[0024]

[Means for Solving the Problem] In order to solve the above problem, in the first aspect of the present invention, when a file is controlled by using real file management information which, with respect to real data serving as data recorded on a recording medium, manages the data as real data and virtual file management information which manages a part of the real file as a virtual referring to the real file, information for identifying the virtual file referring to the real file is included in the real file management information.

[0025]

With the above configuration, for example, when the real file is to be deleted, it can be easily decided whether there is a virtual file referring to the real file or not, although there is the virtual file referring to the real file, a configuration which prevents the real file from being erroneously deleted can be easily structured.

[0026]

More specifically, position information of management information of a virtual file referring to the real file and the number of virtual files referring to the real file are included in the real file management information, so that the above problem is solved. A reference relationship between the real file and the virtual files is designed to be recorded as common information and as one file.

[0027]

In the second aspect of the present invention, information for identifying real file to which the virtual file refers is included in the virtual file management information.

[0028]

With this configuration, when a virtual file is deleted, in deletion of a virtual file or the like, when management information of the real file to which the virtual file refers must be updated, the management information of the real file to which the virtual file refers can be easily referred to.

[0029]

More specifically, position information of management information of a real file to which the virtual file refers is included in the virtual file management information, so that the problem is solved.

[0030]

In the third aspect of the present invention, a plurality of files (real files or virtual files) recorded on a recording medium are managed as one program, and information for identifying files to be managed is included in program management information which manages the program.

[0031]

With this configuration, for example, when a file it to be deleted, it can be easily decided whether there is a program including the file or not, and, although there is the program including the file, a configuration which prevents the file from being erroneously deleted can be easily structured.

[0032]

More specifically, in management information of a file, position information of management information of a program including the file or the number of programs including the file are included, so that the problem is solved.

[0033]

In the fourth aspect of the present invention, in management information of a program, information for identifying a file included in the program is included.

[0034]

With the above configuration, in deletion of a program, when management information of a file included in the program must be updated, the management information of the file included in the program can be easily referred to.

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2001-101050
(43)Date of publication of application : 13.04.2001

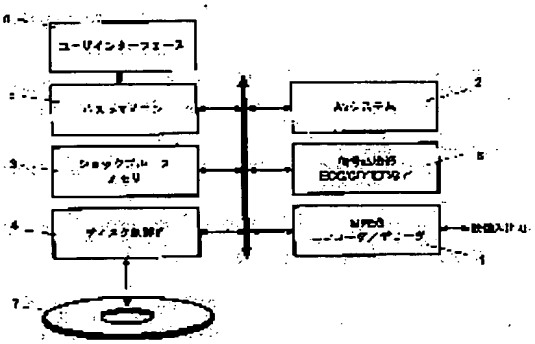
(51)Int.CI. G06F 12/00
H04N 5/76
H04N 5/91

(21)Application number : 11-273913
(22)Date of filing : 28.09.1999

(71)Applicant : SHARP CORP
(72)Inventor : IWANO HIROTOSHI

(54) METHOD FOR MANAGING FILE

(57)Abstract:
PROBLEM TO BE SOLVED: To prevent the invalidity of access due to the absence of the data of a real file being the destination of reference of a virtual file if the real file is simply deleted when the real file is referred to by the virtual file.
SOLUTION: When a file is controlled by using the management information of a real file for managing real data being data recorded on a recording medium as a real file and the management information of a virtual file for managing the data as a virtual file referring to one part of the real file, this method manages files so that information for identifying the virtual file referring to the real file is present in the management information of the real file.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開2001－101050
（P2001－101050A）
(43)公開日　平成13年４月13日(2001.4.13)

(51)Int.Cl.⁷	識別記号	FI		テーマコード(参考)
G06F 12/00	520	G06F 12/00	520P	5B082
			520E	5C052
H04N 5/76		H04N 5/76	B	5C053
5/91		5/91	N	

審査請求　未請求　請求項の数11　ＯＬ　（全 35 頁）

(21)出願番号　特願平11－273913

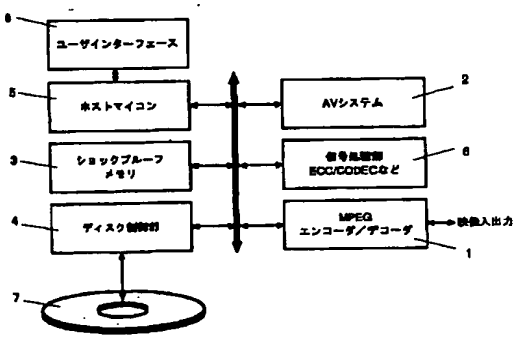
(22)出願日　平成11年９月28日(1999.9.28)

(71)出願人　000005049
シャープ株式会社
大阪府大阪市阿倍野区長池町22番22号
(72)発明者　岩野　裕利
大阪府大阪市阿倍野区長池町22番22号　シャープ株式会社内
(74)代理人　100103296
弁理士　小池　隆彌

Ｆターム(参考)　5B082 AA13 EA01 EA04 EA05 EA07 EA09
5C052 AA01 AC08 CC11 DD07
5C053 FA14 FA23 GB38 HA29 KA01

(54)【発明の名称】　ファイル管理方法

(57)【要約】
【課題】　実ファイルを仮想ファイルが参照している場合、単純に実ファイルを削除してしまうと、仮想ファイルの参照先である実ファイルのデータが無くなりアクセスできなくなるという問題がある。
【解決手段】　記録媒体上に記録されるデータである実データに対応して、該データを実ファイルとして管理する実ファイルの管理情報と、該実ファイルの一部分を参照する仮想ファイルとして管理を行う仮想ファイルの管理情報を用いて、ファイルの制御を行う際に、実ファイルの管理情報中に、該実ファイルを参照している仮想ファイルを識別するための情報を有する。



【特許請求の範囲】

【請求項１】　記録媒体にデータを記録及び再生する装置におけるファイル管理方法であって、
記録媒体上に記録された所定のデータの記録媒体上での領域を示す情報を含み、該データを１つの実ファイルとして管理する第１のファイル管理情報と、
前記実ファイルの全体或いは一部分の記録媒体上での領域を示す情報を含み、該領域のデータを仮想ファイルとして管理する第２のファイル管理情報を前記記録媒体に記録するものであり、
前記第１のファイル管理情報には、当該第１のファイル管理情報により管理されている実ファイルの領域を共有している仮想ファイルを識別するための情報を含むことを特徴とするファイル管理方法。

【請求項２】　前記仮想ファイルを識別するための情報は、該仮想ファイルを管理する第２のファイル管理情報の記録媒体上での位置を示す情報であることを特徴とする前記請求項１に記載のファイル管理方法。

【請求項３】　前記第１のファイル管理情報には、当該第１のファイル管理情報により管理されている実データの領域を共有している仮想ファイルのファイル数を示す情報を含むことを特徴とする前記請求項１あるいは２に記載のファイル管理方法。

【請求項４】　記録媒体にデータを記録及び再生する装置におけるファイル管理方法であって、
記録媒体上に記録された所定のデータの記録媒体上での領域を示す情報を含み、該データを１つの実ファイルとして管理するファイル管理情報を前記記録媒体に記録し、
前記実ファイルの全体或いは一部分の記録媒体上での領域を示す情報を含み、該領域のデータを仮想ファイルとして管理するファイル管理情報を前記記録媒体に記録し、
前記実ファイルと、該実ファイルの領域を共有している仮想ファイルとの共有関係を示す情報を含む共有情報を１つのファイルとして前記記録媒体に記録することを特徴とするファイル管理方法。

【請求項５】　前記実ファイルの削除要求があった場合に、該実ファイルとともに、該実ファイルの領域を共有している仮想ファイルを削除することを特徴とする前記請求項１乃至４のいずれかに記載のファイル管理方法。

【請求項６】　前記実ファイルの削除要求があった場合に、該実ファイルの領域のうち、仮想ファイルによって、共有されていない該実ファイルの領域のみを削除することを特徴とする前記請求項１乃至４のいずれかに記載のファイル管理方法。

【請求項７】　記録媒体にデータを記録及び再生する装置におけるファイル管理方法であって、
記録媒体上に記録された所定のデータの記録媒体上での領域を示す情報を含み、該データを１つの実ファイルとして管理する第１のファイル管理情報と、前記第１のファイル管理情報により管理されている実ファイルの全体或いは一部分の記録媒体上での領域を示す情報を含み、該領域のデータを仮想ファイルとして管理する第２のファイル管理情報を前記記録媒体に記録するものであり、
前記第２のファイル管理情報には、当該第２のファイル管理情報により管理されている仮想データの領域が共有している実ファイルの第１のファイル管理情報の記録媒体上での位置を示す情報を含むことを特徴とするファイル管理方法。

【請求項８】　記録媒体にデータを記録及び再生する装置におけるファイル管理方法であって、
記録媒体上に記録された所定のデータの記録媒体上での領域を示す情報を含み、該データを１つのファイルとして管理するファイル管理情報を前記記録媒体に記録し、
前記ファイル管理情報により管理されている複数のファイルを識別するための情報を含み、該複数のファイルを１つのプログラムとして管理するプログラム管理情報を前記記録媒体に記録するものであり、
前記ファイル管理情報には、当該ファイルを含むプログラムを識別するための情報を含むことを特徴とするファイル管理方法。

【請求項９】　前記ファイルを含むプログラムを識別するための情報は、該プログラムを管理するプログラム管理情報の記録媒体上での位置を示す情報であることを特徴とする前記請求項８に記載のファイル管理方法。

【請求項１０】　前記ファイル管理情報には、当該ファイルを含むプログラムの数を示す情報を含むことを特徴とする前記請求項８に記載のファイル管理方法。

【請求項１１】　記録媒体にデータを記録及び再生する装置におけるファイル管理方法であって、
記録媒体上に記録された所定のデータの記録媒体上での領域を示す情報を含み、該データを１つのファイルとして管理するファイル管理情報を前記記録媒体に記録し、
前記ファイル管理情報により管理されている複数のファイルを識別するための情報を含み、該複数のファイルを１つのプログラムとして管理するプログラム管理情報を前記記録媒体に記録するものであり、
前記プログラム管理情報には、当該プログラムに含まれるファイルの管理情報の前記記録媒体上での位置を示す情報を含むことを特徴とするファイル管理方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、記録媒体にファイルを記録する場合におけるファイルの管理方法に関するものである。

【０００２】

【従来の技術】近年のマルチメディアの普及に伴い、映像、音楽、静止画などの様々なマルチメディアデータを、記録媒体へ記録する需要が高まってきている。記録

媒体の中でも、従来はビデオテープやオーディオテープなどのテープメディアが主流であったが、近年はハードディスク、光磁気ディスクなどのディスクメディアに記録することが多くなってきている。テープメディアの場合は、テープの先頭から順番に記録再生を行なうシーケンシャルアクセスを前提とした記録メディアであり、ランダムアクセス性には優れていない。

【０００３】例えば、ビデオテープで、ある特定の箇所から再生を開始したい場合には、テープをその箇所まで早送りや巻戻しをして再生する必要がある。目的の箇所を示すインデックス情報が予め設定されていれば、頭出し操作1つで操作は終るが、目的の箇所までテープを物理的に送らなければならない。また、インデックス情報が無い場合は、再生しながらの早送りをして目的の箇所を探したり、見当をつけて早送りを行ない最終的に再生を行ない目的の箇所を探す必要があった。このように、テープメディアを用いた場合、物理的なテープの移動が伴うのでランダムアクセスに不向きであった。

【０００４】一方、ディスクメディアにおいては、ランダムアクセス性に優れておりテープメディアと比較した場合、任意の箇所にアクセスするためのアクセス時間は無視できるレベルのものである。よって、ディスク上のどこにデータがあっても、瞬時にアクセスすることが可能である。

【０００５】ディスクにデータを記録する場合、一般的に記録したデータがディスクのどこに記録されているかを管理するための管理情報が必要である。これらの管理手法を提供してくれるものとして、広くMS-DOSやWindowsなどで使われているJIS X 0605-1990、通称FATシステムやDVDなどで使われているOSTA(Optical Storage Technology Association)によるUDF (Universal Disk Format) などがあげられ、ディスクの論理ファイルシステムと呼ばれている。

【０００６】図41に示すように、論理ファイルシステム23を用いる事によって、論理ファイルシステムの管理情報25であるファイル名とディスク26上の実データ24を関連付けることができ、ユーザシステム20がファイル名を指定する事によってディスク26上の実データ24にアクセスすることが可能となる。また、論理ファイルシステムの管理情報25であるディレクトリの概念を用いることによって、ファイルの階層構造を表現することが可能となる。

【０００７】ファイルやディレクトリなどを管理する論理ファイルシステムの管理情報25は、ディスク26上に記録される。一般的に論理ファイルシステム23の仕様に合せてディスクにアクセスするためのデバイスドライバ22が用意される。このデバイスドライバ22を使うことによって、ディスク26にアクセスするユーザシステム20はファイル名を指定してOPEN、WRITE、READ、DELETE、COPY、MOVEなどと言った抽象化されたファイル処理21コマンドによってディスク26にアクセスすることが可能となる。

【０００８】これらの抽象化されたコマンドを受け取ったデバイスドライバ22は、ディスク26に記録された論理ファイルシステムの管理情報25を元にディスクアクセスを行う。このように、論理ファイルシステムの管理情報25はデバイスドライバ22がアクセスする情報であり、ユーザシステム20がアクセスする必要がない情報となる。

【０００９】ディスクのランダムアクセス性を利用すると、1つのファイルに対応するデータであってもディスク上で連続的に記録されている必要はない。つまり1つのファイルで管理する一連のデータがディスク上で分断して記録されていても、ディスクからそれらの分断を順番に読み出して行くことによって、各分断点において次にデータを読み出すディスク上の位置までディスク装置のヘッドを移動させるシークが発生するがテープメディアと比較して無視できるデータ読み出し中断時間なので、あたかも連続的に対応するデータをディスクから読み出しているのと同様の効果が得られる。

【００１０】このようにランダムアクセス性に優れたディスクメディアにおいて、例えば動画データのようなマルチメディアデータを記録する場合について説明を行なう。説明の都合上、ユーザによって記録された記録開始から終了あるいは一時停止などの一連の映像データをオリジナルシーンと呼び、オリジナルシーンの任意の箇所を選択した映像データの管理単位をユーザシーンと呼ぶ事とする。また、いくつかのオリジナルシーンを組み合わせた管理単位をオリジナルプログラムと呼び、いくつかのユーザシーンを組み合わせた管理単位をユーザプログラムと呼ぶ事とする。

【００１１】映像データをディスクに記録する際、ユーザによる記録開始から終了までの映像を1つのオリジナルシーンとして管理するが、1つのオリジナルシーンはユーザにとって一連の映像データシーケンスであり、オリジナルシーン単位でディスク上の映像データを管理する事は好都合である。そこで、前述の論理ファイルシステム利用して１つのオリジナルシーンを１つのファイルとして管理を行なう事とする。よって、ディスクに記録したオリジナルシーンが追加になる度に、ファイルという管理単位が追加されて行く。

【００１２】映像データを再生する場合は、再生を行ないたい映像データを管理しているファイル名をデバイスドライバに指定することによって、ディスクから映像データを読み出すことが可能となる。ディスクのランダムアクセス性を活かし、一連のオリジナルデータがディスク上で連続的に配置されている必要がなく、このディスクのランダムアクセス性は、編集においても効果を発揮する。

【００１３】ディスクメディアに記録された映像データを編集する場合について説明する。ビデオカメラなどを

考えた場合、撮影したオリジナルシーンの映像データは必ずしも全て必要な映像であるとは限らない。例えば、オリジナルシーンの最初に不要な映像が映っていたりと不要な部分が含まれている事も考えられる。編集操作を行なうことによって、必要な任意の箇所をオリジナルシーンから選択してユーザシーンとして定義することが可能である。従来のテープメディアにおいては、一般的にユーザシーンを組み合わせた編集結果であるユーザプログラムを別のテープメディアなどにコピーを行なっていた。

【００１４】しかしディスクメディアの場合は、ディスク上の任意の箇所にアクセスするためのアクセス時間がテープメディアの場合と比較して極めて短いため、素材データであるオリジナルシーンのデータをコピーしたり手を加えることなくディスク上のデータを共有する形で、ユーザシーンやユーザプログラムの再生が可能となる。ユーザによって定義されたユーザシーンが参照しているシーンのディスク上での位置（範囲）を示す情報さえあれば、ユーザが定義した任意の再生順序であるユーザシーンにアクセスする事が可能となる。

【００１５】特開平10-64247における公報において、前述のオリジナルシーンが記録されたディスク上の位置情報に関してファイルで管理を行ない、ディスク上のオリジナルシーンのデータを共有した形で、ユーザによって定義されたユーザシーンが参照しているオリジナルシーンとその範囲を示す情報を、仮想ファイルとして論理ファイルシステムの管理情報として持つ編集装置および記録媒体について説明している。

【００１６】図42に示すように3つのオリジナルシーンがディスク上に記録されており、それぞれのオリジナルシーンにおいて任意の箇所を選択し3つのユーザシーンが定義されている例について説明する。この公報によると、ディスク上のオリジナルシーンのデータは論理ファイルシステムによって、ファイル名OS0001.MPG、OS0002.MPG、OS0003.MPGとディスク上の記録位置が関連付けられ管理されている。

【００１７】一方、ユーザシーンは論理ファイルシステムによって仮想ファイルとしてUS0001.MPG、US0002.MPG、US0003.MPGという名前で管理されている。仮想ファイルの論理ファイルシステムの管理情報は参照しているオリジナルシーンのファイルを特定するための情報とデータの選択箇所を示す開始点と長さの集合である。仮想ファイルのポインタ情報によってオリジナルシーンの任意の箇所を抜き出し、仮想的なファイルとして扱うことが可能となる。

【００１８】ファイルや仮想ファイルの管理情報は論理ファイルシステム階層のものであるため、前述したようにこのディスクにデバイスドライバを介してアクセスすることによって、ディスク上にオリジナルシーンのデータが記録されているだけにも関わらず、ユーザシステムから見ると仮想ファイルで管理されるユーザシーンに対応するデータが仮想的に別途ディスクに記録されているものとして扱われる。つまり、仮想ファイルUS0001.MPGをディスクから読み出す命令をデバイスドライバに渡すことによって、ユーザシーンを構成するオリジナルシーン中の選択箇所のデータが自動的に読み出されることになる。

【００１９】

【発明が解決しようとする課題】上述した特開平10-64247号公報に記載の発明において、ディスク上で記録されたオリジナルシーンをファイルで管理し、ディスク上のオリジナルシーンのデータを参照する形で定義されるユーザシーンを仮想ファイルとして管理を行なうことができる。

【００２０】しかしながら、オリジナルシーンを管理しているファイルとユーザシーンを管理する仮想ファイルの参照関係を示す情報が用意されていない。例えば、オリジナルシーンを管理する1つのファイルを複数のユーザシーンが参照している場合を想定する。ここで、仮にオリジナルシーンを削除しようとする場合、単純にオリジナルシーンを削除してしまうと、ユーザシーンを管理している仮想ファイルの参照先であるオリジナルシーンのデータが無くなりアクセスできなくなるという問題がある。

【００２１】よって、オリジナルシーンのデータを完全に削除したり、部分的に削除したりする前には、削除しようとするオリジナルデータを参照している仮想ファイルがあるかどうか、定義されている全ての仮想ファイルについて、定義されている仮想ファイルの参照先が、これから削除しようとする部分に該当するか否かを調べる必要がある。

【００２２】ユーザシーンを管理する仮想ファイルが数個であれば大した手間はかからないが、仮想ファイルの数が多くなると、参照しているかどうかを調べるのに手間がかかると言う問題点を有している。

【００２３】また、同様にオリジナルシーンを削除する際に例えば、オリジナルシーンを参照しているユーザシーンも一緒に削除するといった処理を行う場合や、ユーザシーンから参照されている部分を残して参照されていない箇所を削除するといった処理を行う場合に、上記の問題点と同様に参照関係を把握するのに手間がかかるという問題点を有している。

【００２４】

【課題を解決するための手段】上記課題を解決するために、本発明の第１の発明においては、記録媒体上に記録されるデータである実データに対応して、該データを実ファイルとして管理する実ファイルの管理情報と、該実ファイルの一部分を参照する仮想ファイルとして管理を行う仮想ファイルの管理情報を用いて、ファイルの制御を行う際に、実ファイルの管理情報中に、該実ファイル

を参照している仮想ファイルを識別するための情報を有している。

【００２５】このように構成することによって、例えば実ファイルを削除しようとした場合に、該実ファイルを参照している仮想ファイルがあるか否かを容易に判断することが可能となり、誤って、参照している仮想ファイルがあるにもかかわらず、実ファイルを削除してしまうということを防止するための構成を容易に構築できる。

【００２６】具体的には、実ファイルの管理情報中に、該実ファイルを参照している仮想ファイルの管理情報の位置情報や、該実ファイルを参照している仮想ファイルの数を備えることによって、上記課題を解決するものである。また、該実ファイルと仮想ファイルの参照関係を共有情報として一つのファイルとして記録するように構成できる。

【００２７】本発明の第２の発明においては、仮想ファイルの管理情報中に、該仮想ファイルが参照している実ファイルを識別するための情報を有している。

【００２８】このように構成することによって、仮想ファイルを削除した場合など、該仮想ファイルが参照している実ファイルの管理情報を更新する必要が生じた場合に、容易に該仮想ファイルが参照している実ファイルの管理情報を参照することができる。

【００２９】具体的には、仮想ファイルの管理情報中に、該仮想ファイルが参照している実ファイルの管理情報の位置情報を備えることによって、上記課題を解決するものである。

【００３０】本発明の第３の発明においては、記録媒体上に記録された複数のファイル（実ファイル或いは仮想ファイル）を１つのプログラムとして管理し、該プログラムを管理するプログラム管理情報に、管理するファイルを識別する情報を備えている。

【００３１】このように構成することによって、例えば、ファイルを削除しようとした場合に該ファイルを含むプログラムがあるか否かを容易に判断することが可能となり、誤って、該ファイルを含むプログラムがあるにもかかわらず、該ファイルを削除してしまうということを防止するための構成を容易に構築できる。

【００３２】具体的には、ファイルの管理情報中に、該ファイルをプログラムに含むプログラムの管理情報の位置情報や当該ファイルを含むプログラムの数を備えることによって、上記課題を解決するものである。

【００３３】本発明の第４の発明においては、プログラムの管理情報中に、該プログラムに含まれるファイルを識別するための情報を有している。

【００３４】このように構成することによって、プログラムを削除した場合など、該プログラムが含むファイルの管理情報を更新する必要が生じた場合に、容易に該プログラムが含んでいるファイルの管理情報を参照することができる。

【００３５】

【発明の実施の形態】以下、本発明のファイル管理方法に関する実施形態について、図面を用いて詳細に説明する。本実施形態において、記録装置として携帯型のディスクを用いたビデオカメラを、ディスクに記録する映像データはMPEGを、そして編集方法に関しては断りがない限りオリジナルの素材をコピーしたり変更を加えることを行なわずにユーザシーンやユーザプログラムを作成する非破壊編集を想定するものである。また、ディスク装置に関しては、据え置き型のビデオデッキや、記録媒体はハードディスクや半導体メモリであっても本実施形態をそのまま適用できるものである。

【００３６】以下、本実施形態の説明において、ディスクに記録されたオリジナルシーンに対応するディスク上のオリジナルデータのファイルを実ファイルと呼び、オリジナルデータを参照する形で構成されるユーザシーンやユーザプログラムを管理するファイルを仮想ファイルと呼び、区別して説明するものとする。

【００３７】図1に本発明の一実施形態におけるファイル管理方式が対象とする記録再生装置の機能ブロック図を示す。図中において、MPEGエンコーダ／デコーダ1は、MPEGデータをエンコードしたり、デコードする部分であり、AVシステム部2は、記録時には、MPEGエンコーダから得られたMPEGデータとオーディオデータとを、ディスクに記録する際のストリームデータにするために多重を行なったり、ヘッダ情報などの付加を行ない、再生時は、逆にディスク7から読み出したストリームデータから、再生しようとする映像とオーディオデータとを取り出し、MPEGデコーダに渡す部分である。

【００３８】ショックプルーフメモリ3は、色々な処理を行う上で一時的にデータを保管する目的や、ストリームデータを格納し、ディスクドライブがシークを行なっているなどの理由で、実際にデータを読み出したり書き込んだりすることができないときであっても、記録再生に支障をきたすことを防ぐ役割を担っている。信号処理部6は、ECC（エラー訂正符号）を記録するデータに付加したり、再生するデータのエラー訂正処理を行ったり、データをディスクに記録する際にセクタにデータを記録できるような形に整えるセクタコーデックなどを行なう部分である。これらの処理はショックプルーフメモリ3に格納されているデータに対して行う。

【００３９】ディスク制御部4は、サーボをコントロールしたり、ディスクアクセスを制御する部分である。ホストマイコン5は、本システム全体を制御する部分で、各処理部に対して制御信号を出したり、受けたりすることによって、制御を行なう。またユーザからの指示はユーザインターフェース8によって受け付け、ユーザからの要求をホストマイコン5で実際に制御する。

【００４０】AVデータであるMPEGストリームをディスクに記録する場合について説明をする。コンピュータ用途

のデータの場合と異なり、AVデータをディスクに記録したり再生したりすることを考えると、ある決まった時間以内にデータの読み書きが行なえる事が重要になってくる。例えば、これから再生しなければならないデータがディスクから読み出せていない場合、再生画面が途切れてしまうことが起こり得る。読み出すべきAVデータがディスク上で連続的に配置されている場合は、このような状況に陥る可能性は低いが、読み出すべきAVデータがディスク上で分断して記録されている場合は、各分断点において次にアクセスすべき箇所にディスクのヘッドが移動するためにシークが発生する。シークが発生するということは、シーク中ディスクからのデータの読み込みが中断することを意味する。

【００４１】一般的にシークや外的なショックなどによるデータの読み込み中断があっても再生画面が途切れないようにするために、AVデータの再生レートがディスクからのデータの読み出しレートより低いことを利用して、読み出したAVデータを一度ショックプルーフメモリに格納し、ディスクからのデータの読み出しが一時的に中断してもショックプルーフメモリに蓄えられた余裕分を使うことによって再生画面が途切れない様にしている。しかし、ショックプルーフメモリへのデータの流入が長い間止まったり、中断時間が短くても何回も止まったりすると再生画面が途切れてしまうことがある。一番確実に再生画面を途切れなく再生するシームレス再生を実現する方法として、データをディスクに連続的に書き込むことが挙げられる。連続的にデータが記録されていれば、そのデータを読み出す際にアクセス途中でシークが発生することは基本的に無いからである。

【００４２】連続的に書き込みが行なえないような状況がどのような事であるかを考えてみると、例えばオーディオデータや静止画像データなどのようにAVデータとは種類の異なるデータが同一領域に混在する場合と、AVデータを削除したディスクの任意の空き領域に、新たにAVデータを記録する場合などが考えられる。前者の場合は、AVデータを記録する領域を定義し他の種類のデータが同一領域に記録されないようにする事によって、問題は解決される。

【００４３】そこで本実施形態で説明する論理ファイルシステムでは、ディレクトリとしてディスク上の連続領域を管理する機能を有している。通常の論理ファイルシステムにおいてディレクトリは概念的な枠組であるのに対して、ディスク上の連続領域を管理しそのディレクトリの下に作成されるファイルがその確保された連続領域内に書き込まれることを保証する機能を提供している。逆にこの機能によると、この連続領域を確保したディレクトリの下位階層以外で作成されたファイルは、この連続領域に書き込まれないことを保証するものである。

【００４４】図2に、本論理ファイルシステムに従ってAVデータを後述するAV Areaに記録した場合の例を示す。MPEGデータであるAVデータを記録する領域として連続領域を確保したディレクトリORIGINALを用意する。そのディレクトリの下にユーザによって撮影されたオリジナルシーンをファイル単位でOS0001.MPG、OS0002.MPG、OS0003.MPGのように記録していく。

【００４５】連続確保した領域内で、先頭から順番にオリジナルシーンを記録していくため、削除を行なわない限りディスク上のデータは連続的に配置されることになる。当然ながら、図の例のようにファイルを管理する論理ファイルシステムの管理情報である後述するFile Descriptorでは、一連のデータであってもディスク上で分断して記録されている場合にも対応している。図の例の場合は、OS0001.MPGはディスク上で、図中のEUS0-1とEUS0-2のように分断されて記録されている。

【００４６】このようにオリジナルシーンは連続領域を確保しているディレクトリの下に実ファイルとして管理される。例えば、記録した順番がわかるようにこのオリジナルシーンの実ファイルに名前を付けることができる。従来のテープメディアにおける再生のように、テープの最初から撮影した順番に再生を行なうには、オリジナルシーンの１番から順番に再生を行なって行く事によって同様の効果が得られる。

【００４７】本実施形態においては、素材データであるオリジナルシーンのデータをコピーしたり手を加えることなくディスク上のデータを共有する形で、オリジナルシーンの任意の箇所を選択することによって定義されるユーザシーンを論理ファイルシステムの仮想ファイルで管理を行うこととする。既にディスク上でファイルとして管理されているデータの任意の箇所を共有するため仮想ファイルと呼び、ディスク上の実データ（オリジナルシーン）を管理する実ファイルと区別する。

【００４８】ここで、ディスク上で実ファイルとして管理されているオリジナルシーンに対応するデータを複数のユーザシーンが参照している場合を考える。ここで、仮にオリジナルシーンを管理している実ファイルを完全あるいは部分的に削除したりする場合、ユーザシーンを管理している仮想ファイルが参照しているオリジナルシーンのデータが無くなりアクセスできなくなるため、相互の参照関係を示す情報が必要となる。

【００４９】このように、オリジナルシーンを管理する実ファイルとユーザシーンを管理する仮想ファイルの参照関係を示す情報を用意することによって、オリジナルシーンのデータを参照している仮想ファイルに影響をおよばさないように削除等をすることが可能となる。

【００５０】次に、本実施形態で扱うMPEGストリームの構成の一例について説明を行なう。図３のストリーム構成において、EUS（Editable Unit Sequence）は、複数のEU（Editable Unit）によって構成され、REC Start（記録開始）からRec Stop（記録停止）或いはRec Pause（記録一時停止）に対応する単位である。上記図２に

示すように、１つの実ファイルがEUSに対応する。

【００５１】尚、EUは破壊編集における最小単位である。破壊編集とは、ディスク上での移動や削除を伴う編集のことを意味し、破壊編集の最小単位とは、ディスク上での移動や削除がEU単位でしか行うことができないことを意味する。 EUは1つ以上のVU（Video Unit）及び1つのPRU（Post Recording Unit）によって構成され、ディスク上では必ず連続的に記録されなけらばならない。尚、PRUが無いストリーム構成もある。なおPost Recordingとはアフレコのことを意味する。

【００５２】PRUのディスク上での開始位置及び終了位置は、ECCブロックの境界でなければならないという制限がある。また、PRUはEU内のビデオデータと同期して再生するPost Recording用のデータ領域であるので、最低でもEUのビデオデータの提示時間に相当するだけのデータが記録できる領域がなければならない。また、VUはUnit Headerと１GOP以上の映像データ及び対応する音声データとをまとめた単位である。

【００５３】前記EUSを2048byteの固定長のブロックに分割を行なう。1つのブロックは1つの論理ブロックに格納され、1つのブロックは原則として1個のパケットで構成される。ここでのパケットは、ISO/IEC13818-1で規定されるPES Packetに準拠し、ディスクにはこのパケットを記録していくことになる。

【００５４】図4にEUSとブロックとの関係を示す。図中において、PRUはUH BLK（Unit Header Block）、A BLK（Audio Block）、P BLK（Padding Block）で構成される。UHBLKは、ＰＲＵに関するヘッダ情報を格納したパケット、A BLKは、ISO/IEC13818-3で規定されるオーディオパケット、P BLKは、ISO/IEC13818-1で規定されるパディングパケットがそれぞれ格納される。また、VUはUH BLK（Unit Header Block）、A BLK（Audio Block）、V BLK（Video Block）によって構成される。UH BLKは、VUに関するヘッダ情報を格納したパケット、A BLKは、ISO/IEC13818-3で規定されるオーディオパケット、V BLKは、ISO/IEC13818-2で規定されるビデオデータを格納したパケットがそれぞれ格納される。

【００５５】PRUの領域は、初期状態などPost Recording Dataが存在しない場合、前記ヘッダブロックのUH BLK以外は、パディングブロック(P BLK)でパディングされる。Post Recordingされると、A BLKなどのように、オーディオブロックなどが実際に記録される。このオーディオデータは、対応するVU内のビデオデータと同期して再生されるものである。

【００５６】VUはオーディオ部分が複数のA BLKによって構成され、ビデオデータ部分は複数のV BLKによって構成される。このオーディオデータは、ビデオデータと同期して再生されるものである。

【００５７】上記のMPEGストリームをディスクに記録する際の実ファイルで管理されるオリジナルシーンと仮想ファイルで管理されるユーザシーンの関係に関して説明を行なう。

【００５８】MPEGデータは、フレーム内符号化画像（Iピクチャ）、フレーム間順方向予測符号化画像（Pピクチャ）、双方向予測符号化画像（Bピクチャ）という3種類の画像圧縮手法を使って、効率的にデータ量を削減している。Pピクチャ及びBピクチャは、Iピクチャに基づいて生成されているため、そのデータだけではデコードすることができない。

【００５９】MPEGデータを先頭から順番にデコードして再生する場合には、問題は生じないが、MPEGデータの途中から再生したり、任意のフレームだけを拾って再生するといった特殊再生を行なう場合には、問題が生じる。それは、再生を開始したい対象となるフレームが、PピクチャやBピクチャの場合、実際にそのフレームをデコードするためには、レファレンスとなったIピクチャやPピクチャのデータがないと、デコードできないからである。

【００６０】このような問題を解決するために、MPEGにおいては、何枚かのフレームを集めてGOP（Group of Pictures）という構造が用意されている。このGOP構造は、GOPの中には少なくとも1枚のIピクチャがなければならないというものである。

【００６１】従って、GOP構造単位でアクセスを行なえば、そのGOPの中に含まれている各Pピクチャ及びBピクチャのレファレンスとなるIピクチャが含まれているので、目的のフレームをデコードすることが保証される。

【００６２】MPEGデータを対象にランダムアクセスを行なう場合は、GOP構造単位で行なう必要がある。例えば、GOP構造の途中のフレームから再生を行ないたい場合であっても、GOP単位でデータをデコードした上で、目的のフレームからを実際に表示するように制御すれば、そのフレームから再生を開始した事と等価になる。

【００６３】このようなMPEGの特性を考慮すると、ユーザによって選択されているオリジナルシーンの任意の箇所で構成されるユーザシーンに対応するディスク上でのデータは、前述のGOPの集合であるVU単位で行われなければならない。

【００６４】図5にオリジナルシーンである実ファイルと、ユーザシーンである仮想ファイルとの関係を示す。この例では、オリジナルシーンは実ファイルOS0001.MPGであり、該ファイルの管理情報で管理されている。OS0001.MPGは、ディスク上で3つの連続領域に分断されて記録されている。オリジナルシーンをディスクから読み出す際は、ユーザシステムがデバイスドライバにファイルOS0001.MPGの読み出しを指定することによって、論理ファイルシステムの管理情報で管理されるディスク上の分断1A、分断2A、分断3Aの位置情報（開始アドレスと長さ）を元に、デバイスドライバが分断1A、分断2A、分断3Aの順番にディスクからデータを読み出す。

【００６５】ここでオリジナルシーンを編集してVU#1からVU#7までを選択して定義されたユーザシーンについて説明する。ユーザシーンは仮想ファイルUS0001.MPGとして管理されており、論理ファイルシステムの管理情報内の分断1B、分断2B、分断3Bの位置情報（開始アドレスと長さ）によってディスク上のオリジナルデータを参照している。ここで、ディスク上にはVU#0からVU#9に対応する映像データが1つしか記録されていないが、仮想ファイルで参照される任意箇所の位置情報によって、部分的なデータの読み出しがユーザシステムにおいては仮想ファイルを指定することによって可能となる。

【００６６】このように、仮想ファイルはディスク上に記録されているオリジナルシーンのデータを参照するものである。また、仮想ファイルは同一のオリジナルシーンを複数のユーザシーンで参照することも可能である。図6に2つのユーザシーンを管理する仮想ファイルの様子を示す。この例では、VU#0からVU#9までがオリジナルシーンであり、VU#1からVU#7までがユーザシーン1（US0001.MPG）、VU#6からVU#9までがユーザシーン2（US0002.MPG）という構成である。つまり、VU#6,VU#7が2つのユーザシーンで参照されていることになる。

【００６７】次に、本実施形態において使用する論理ファイルシステムについての詳細を説明する。論理ファイルシステムとは、ディスクメディアに記録されているデータをファイル形式で管理をすることによって、ディスクを使用するユーザやアプリケーションプログラムにとって使い勝手の良さを提供するものである。ディスク上の任意の箇所に分断されて記録されているようなデータであってもファイルを指定することによって、データの読み出しが容易に行なえることが可能となる。また、ディレクトリの概念によってディスクに記録されるファイルを効率良く整理し管理することが可能となる。

【００６８】ディスクは、一般的にセクタの集合で構成されておりセクタにはアドレスが付加されている。特に論理ファイルシステムが管理を行なえるセクタを論理セクタと呼ぶ事とする。通常ディスクには、パリティーを記録する領域やセクタに欠陥などが生じた時に、代替として使用するセクタの領域など、論理ファイルシステムがアクセスできない領域がある。また、本発明では更に論理ブロックという概念を導入する。論理ブロックとは2n（n=>0）個の論理セクタを集めた管理単位である。本論理ファイルシステムにおいての最小管理単位をこの論理ブロックとする。本説明においては論理セクタの大きさは2KBとし、論理ブロックの大きさも同じ2KBとする。

【００６９】本実施形態で説明する論理ファイルシステムにおいては、Volumeレベルの論理ファイルシステムの管理情報とAreaレベルの論理ファイルシステムの管理情報に区別されている。図１１に記録媒体上での様子を示す。Volumeレベルの論理ファイルシステムの管理情報によって、特定用途のAreaを任意の個数定義することが可能となる。この図１１の例では、記録媒体上に１つのAreaを定義している場合を示している。つまり記録媒体全体がVolumeとなる。

【００７０】このVolumeのうち、Volumeを管理する領域がManagement Spaceであり、それ以外の部分がAllocatable Spaceである。この例では１つのVolumeに１つのAreaを定義するので、このAllocatable Space = Areaとなる。

【００７１】Volumeを管理する管理情報は Volume Management Spaceに記録される。１つのVolumeにおいて、このVolume Management Spaceを除く領域が Area となる。このVolume Management Spaceには、ディスク全体の基本管理情報を記録するPrimary Volume Descriptor (PVD)、エリアの基本管理情報を記録するArea Descriptor (AD)が記録される。

【００７２】前述のPVDはディスクの論理セクタ番号0（LSN=0）と論理セクタ番号16（LSN=16）に同一内容のものが記録される。論理セクタ16に記録されるPVDはバックアップ目的で使われる。論理セクタ0に記録されたPVDにアクセスすることによって、ディスクの基本管理情報とディスク上に定義された特定用途向けのAreaを管理するArea Descriptorが記録されている位置を把握することができる。Volume Management Spaceとして確保された領域の論理セクタ番号32（LSN=32）以降にArea Descriptorが記録される。Area Descriptorは定義されたAreaの数だけVolume Management Spaceに記録される。PVDの情報を元に、Area Descriptorの管理情報をディスクから読み出す。Area Descriptorにより、Areaの基本情報および、Area内で最初にアクセスすべきディスク上の管理情報の位置情報を把握することが可能となる。

【００７３】まず、Primary Volume Descriptor (PVD)のデータ構造について図９を用いて説明する。Primary Volume Descriptor (PVD) は、ディスク全体の基本管理情報を記録する管理記述子である。PVDは、PVDを識別するためのID (Header ID)、管理を行なうディスクの種別 (Disk Type)、ディスクを区別するためのID (Disk ID)、ディスクの大きさ (Volume Size)、Allocatable Spaceにおける空き領域の大きさ (Free Logical Sectors in Allocatable Space)、ディスク名 (Volume Name)、PVDの作成および修正日時 (Creation Time & Date、Modified Time& Date)、Areaを管理するための管理領域であるVolume Management Spaceのディスク上での位置情報である開始論理セクタ番号と論理セクタ数 (Location ofVolume Management Space)、Areaを定義するためのディスク領域を管理するAllocatable Spaceのディスク上での位置情報である開始論理セクタ番号と論理セクタ数 (Location of Allocatable Space)、Allocatable Spaceに定義されたArea数 (Number of Areas)、定義されたAreaの基本情報が記録されるArea Descripto

rのディスク上での記録位置（Location of Area Descriptor）によって構成される。なお、Location of Area DescriptorはNumber of Areasの値だけ存在することになる。つまり、図１１の例のように、１つのVolumeに１つのAreaしかない場合は、Location of Area Descriptorは１つとなる。

【００７４】ここで、図９の表中の、BPはByte Positionを意味し、先頭から見た対応する管理項目の開始位置を示す情報で、Lengthはその管理項目の大きさをByteで表し、Field Nameは管理項目名、Contentsは、管理項目がどのような形式で記録されなければならないかということを示す。Contentsで用いられているデータ型のうち、Uint8は符号無し8bit整数、Uint16は符号無し16bit整数、Uint32は符号無し32bit整数、Uint48は符号無し48bit整数を意味する。Stringは文字列を格納するためのデータ型、Timestampは日時情報を格納する型である。

【００７５】また、contents中の adr#longは、図7に示すように、本論理ファイルシステムの管理記述子で共通に利用するデータ型で、主にディスク上のデータなどの配置を開始点（Location）と長さ（Length）で表現をする。同様に、Header IDは、図8に示すように、本論理ファイルシステムの管理記述子で共通に利用されるデータ型で、論理ファイルシステムの種類（Standard ID）、論理ファイルシステムのバージョン（Standard Version）、管理記述子の種類（Descriptor ID）、管理記述子をディスクに書き込んだメーカ（Manufacturer ID）で構成されている。

【００７６】つぎに、Area Descriptor（AD）のデータ構造について説明する。図10に示すArea Descriptor（AD）は、Areaの基本管理情報を記録する管理記述子である。ADは、ADを識別するためのID（Header ID）、Areaの種類を示す情報（Area Type）、Area内の論理ブロックサイズ（Logical Block Size）、Areaの大きさ（Area Size）、Areaの名前（Area Name）、ADの作成および修正時刻（Creation Time &Date、 Modified Time & Date）、Area内の最初にアクセスすべき管理情報の位置情報である論理ブロック番号（Location of Main Primary Area Descriptor(PAD)）、Area内の最初にアクセスすべき管理情報のバックアップが存在する時の位置情報である論理ブロック番号（Location of Secondary Primary Area Descriptor(PAD)）、Areaのディスク上で分断数（Number of Area Extents）、分断毎のディスク上での位置情報を示す開始論理セクタ番号と論理セクタ数（Locationof Area Extent）で構成される。Location of Area ExtentはNumber of Area Extentの値だけ存在する事になる。例えば、Areaが２つの分断で構成されていれば（Areaが２つの領域からなっていれば）、Location of Area Extentの項目は２つあり、それぞれにそれぞれのAreaの位置情報が記録されることになる。

【００７７】次に、Area内のデータの構成を図１９を用いて説明する。AV Areaは、論理ファイルシステムの管理情報を記録するArea Management Spaceとユーザシステムがデータを記録するExtent Spaceにわかれている。

【００７８】論理ファイルシステムの管理領域であるArea Management Spaceには、Area内の最初にアクセスすべき管理情報であるPrimary Area Descriptor（PAD）、Area内の空き領域を管理するための管理情報であるSpace Management Descriptor、ディレクトリを管理するための管理情報であるDirectory Descriptor、実ファイルを管理するためのFile Descriptor、仮想ファイルを管理するためのVirtual File Descriptorなどが記録されることになる。

【００７９】また、安全性の観点から、Area Management Spaceを2重化することが可能である。2重化を行なう場合は、Area Management SpaceをMainとSecondaryの領域に分割を行なう。それぞれ、MainとSecondaryの領域は同一の内容でなければならない。管理情報を更新する際は両方の領域を更新する必要があり、万が一Mainの管理情報が読み込めなくなった場合、Secondaryの管理領域にアクセスすることによって管理情報の復旧を行なうことが可能である。

【００８０】Areaにアクセスする際には、まずPrimary Area Descriptor（PAD）をディスクから読み出す。PADはAV Area内の論理ブロック番号0に記録されており、Areaの基本情報と共に、管理領域の2重化の有無、空き領域管理情報であるSpace Management Descriptorが記録されているディスク上の開始位置、またルートディレクトリを管理しているDirectory Descriptorが記録されているディスク上の位置を示す情報を備えており、これらのデータを読み出すことが可能となる。

【００８１】図12にPrimary Area Descriptor（PAD）の構成を示す。Primary Area Descriptor（PAD）は、Areaの基本管理情報を記録する管理情報である。PADは、PADを識別するためのID（Header ID）、Areaの大きさ（Area Size）、Area内の管理領域であるArea Management Spaceの空き容量（Free Blocks in Area Management Space）、Area内のデータ領域であるExtent Spaceの空き容量（Free Blocks in Extent Space）、Areaのモードを示す情報（Area Mode）、Area内の論理ブロックサイズ（Logical Block Size）、Areaの名前（Area Name）、PADの作成および修正時刻（Creation Time & Date、Modified Time & Date）、Area内のメインの管理領域の位置情報で有る開始論理ブロック番号と論理ブロック数（Location ofMain Area Management Space）、Area内のバックアップ管理領域の位置情報である開始論理ブロック番号と論理ブロック数（Location of Secondary Area Management Space）、Area内のデータ領域であるExtent Spaceの位置情報である開始論理ブロック番号と論理ブロック数（Location of Extent Space）、Area内の空き領域管理を行なうSpace Management Descriptorの記録

開始位置である論理ブロック番号（Location of Space Management Descriptor）、ルートディレクトリのDirectory Descriptorが記録された位置情報である論理ブロック番号（Location of Root Directory Descriptor）で構成される。

【００８２】図13に、Space Management Descriptor（SMD）の構成を示す。SMDは、Area内の空き領域を管理するための管理記述子である。SMDは、SMDを識別するためのID（Header ID）、１論理ブロックにSMDが格納できずに複数の論理ブロックにまたがって記録される場合に次にアクセスすべき論理ブロック番号（Next Extension）、同様に複数論理ブロックにまたがった場合に１つ前の情報が記録された論理ブロック番号（Previous Extension）、そして空き領域管理を行なう管理情報であるSpace Bitmapで構成される。Space Bitmapとは、Area内の全ての論理ブロックに対して1bitの情報を割り当て、その論理ブロックが使用されている場合はそのbitを１に、未使用の場合は0を記録することによって、Area内の空き領域を管理するためのものである。

【００８３】図14に、Directory Descriptor（DD）の構成を示す。DDはディレクトリを管理するための管理情報である。DDはDDを識別するためのID（Header ID）、1論理ブロックにDDが格納できずに複数の論理ブロックにまたがって記録される場合に次にアクセスすべき論理ブロック番号（Next Extension）、同様に複数論理ブロックにまたがった場合に１つ前の情報が記録された論理ブロック番号（Previous Extension）、ディレクトリの属性情報（Attribute）、ディレクトリ名（Directory Name）、DDの作成および修正時刻（Creation Time & Date、 Modified Time& Date）、管理するディレクトリが含まれるディレクトリのDirectory Descriptorが記録された論理ブロック番号（Pointer to Parent Directory）、ディレクトリ構造でディスク上の連続領域を管理する場合の位置情報である開始論理ブロック番号と論理ブロック数（Location of Contiguous Space）、ディレクトリに含まれるファイルやディレクトリ数（Number of Descriptor Pointer Entries）、ディレクトリに含まれるファイルやディレクトリへのポインタ情報（Descriptor Pointer Entry）で構成される。Descriptor Pointer EntryはNumber of Descriptor Pointer Entriesの値だけ存在することになる。

【００８４】図15にディレクトリに含まれるファイルやディレクトリへのポインタ情報（Descriptor Pointer Entry）の構成を示す。Descriptor Pointer Entryは、ファイルやディレクトリの名前（Entry Name）、管理している情報がファイルかディレクトリかを区別する情報（Entry Type）、ディレクトリあるいはファイルの管理記述子が記録されている位置情報である論理ブロック番号（Location of Descriptor）で構成される。

【００８５】図16にディレクトリの属性情報であるAttributeについて示す。Attributeは16bitの情報であり、Bit0 Read Onlyは管理するディレクトリが読み込み専用であるかを示し、Bit1 Deletedは管理するディレクトリが一時的に削除されたかどうかを示し、Bit2 Contiguousは管理するディレクトリがディスク上の連続領域を確保しているかどうかを示し、Bit3 Allocation Modeはディレクトリの場合は使用しないので、常にZEROを記録する。Bit4、5 CGMS（Copy Generation Management System）は、コピーを許可するか、1世代のみコピーするか、コピーを禁止するかどうかの情報を示す。なお、ディレクトリのAttribute情報は後述するファイルの管理情報であるFile Descriptor内のAttributeと同じ構成を持つ。

【００８６】AV Area内に記録されているファイルやディレクトリはルートディレクトリの管理情報から辿ることによって把握することが可能となる。これらの処理は論理ファイルシステムに従って処理をするデバイスドライバがArea Management Spaceに記録されている論理ファイルシステムの管理情報を参照することによって行うことであり、既に述べているようにユーザシステムは単純に目的のファイルを指定して抽象化されたアクセスコマンドを用いてディスクへのアクセスを行う。

【００８７】本実施形態において、実ファイルは、実際にはその実ファイルを管理するためのArea Management Spaceに記録される論理ファイルシステムの管理情報（後述するFile Descriptor）と、Extent Spaceに記録されるそのファイルに対応する実データから構成されることになる。

【００８８】このFile Descriptor（FD）はファイルを管理するための管理情報である。ファイルに対応する実データがディスク上のどこに記録されているかを管理する情報である。本発明はこのFile Descriptorの構成・構造に特徴を備えるものである。このFile Descriptorは、以下、実施の形態に応じて管理情報の構成が異なるので、詳細は後述する。

【００８９】図17に拡張用の記述子であるExtended Descriptor(ED)について示す。EDはEDを識別するためのID（Header ID）、1論理ブロックにEDが格納できずに複数の論理ブロックにまたがって記録される場合に次にアクセスすべき論理ブロック番号（Next Extension）、同様に複数論理ブロックにまたがった場合に1つ前の情報が記録された論理ブロック番号（Previous Extension）、そして最初の論理ブロックに入り切らなかった情報を格納するスペース（Extended Data）で構成される。

【００９０】説明してきた論理ファイルシステムの管理情報は、ディスクに論理ブロック単位で記録される。論理ブロックの大きさが2KBなので、例えば1つのディレクトリを管理するDirectory Descriptorや1つのファイルを管理するFile Descriptorは最低でもディスク上で2KBを使うことになる。定義したディレクトリに含まれるフ

ファイルやディレクトリが少ない場合は、上記Directory Descriptor内のDescriptor Pointer Entryが少ないので、このDirectory Descriptorは余裕を持って1論理ブロックに収まる。しかし逆にディレクトリに大量のファイルやディレクトリが定義されると、このDirectory Descriptorは1論理ブロックに収まらないために、複数の論理ブロックにまたがる必要がある。このような状況において利用するExtended Descriptorについて、図18に示す。

【００９１】この図では、ある管理記述子が1論理ブロックに収まらず3つの論理ブロックにまたがって記録されている様子である。１つ目の論理ブロックがDirectory Descriptorだとすると、2KBに収まりきらなかったDescriptor Pointer Entryの情報が2つ目以降の論理ブロックに、Extended Descriptorを用いて管理される。このようにExtended Descriptorは先頭にヘッダ情報のみが付いた管理情報を格納するための入れ物であり、Directory Descriptor以外にも複数ブロックにまたがって記録されなければならない管理情報について共通で用いられる。

【００９２】ここで、ユーザがビデオカメラなどを利用して撮影を行なう場合にディスクに記録されるオリジナルシーンの作成手順について説明する。撮影開始から停止あるいは一時停止までの連続的な一連の映像をオリジナルシーンとし、1つの実ファイルとしてディスクに記録する。図2に示したように、ディスク上に記録するオリジナルシーンの実ファイルを格納するために、ルートディレクトリの下に連続領域を確保し管理するContiguous Modeのディレクトリ「ORIGINAL」を作成する。Contiguous Modeのディレクトリなので、このディレクトリの下にファイルを作成しない限り、確保された連続領域にデータが書き込まれることはない。ディレクトリ「ORIGINAL」の下に撮影したオリジナルシーン毎にファイルを作成するが、この時ファイル名を撮影した順番に従ってOS0001.MPG、OS0002.MPG、OS0003.MPGといったように、数字の部分で記録順番が把握できるようにする。オリジナルシーンを撮影順序に従って組み合わせた管理単位をオリジナルプログラムと呼び、ここではディレクトリ「ORIGINAL」に対応する。つまり、従来のテープメディアのようにテープの先頭から最後までを再生するにはディレクトリ「ORIGINAL」を指定し、そのディレクトリの下に保存されているオリジナルシーンのファイルをOS0001.MPGから順番に再生することによって同様の効果が得られる。

【００９３】ユーザシーンを管理する仮想ファイルを管理する論理ファイルシステムの管理情報の構成を実施の形態に応じて説明をする。仮想ファイルを管理するVirtualFile Descriptorを有する第1の実施形態、仮想ファイルを管理するVirtual FileDescriptor、その仮想ファイルを任意に組み合わせて新たなファイル構造を構成する仮想プログラムファイルVirtual Program File Descriptorを有する第2の実施形態、第1の実施形態におけるファイルと仮想ファイルの参照情報をユーザシステムが管理するファイルとして持つ場合の第3の実施形態を説明する。

【００９４】実ファイルを管理するFile Descriptorと仮想ファイルを管理するVirtual File Descriptorを有する第1の実施形態に関して説明する。図20に示すFile Descriptor（FD）は実ファイルを管理するための論理ファイルシステムの管理情報でありデバイスドライバが扱う情報である。FDはFDを識別するためのID（Header ID）、1論理ブロックにFDが格納できずに複数の論理ブロックにまたがって記録される場合に次にアクセスすべき論理ブロック番号（Next Extension）、同様に複数論理ブロックにまたがった場合に１つ前の情報が記録された論理ブロック番号（Previous Extension）、ファイルの属性情報（Attribute）、ファイル名（File Name）、FDの作成および修正時刻（Creation Time & Date、Modified Time &Date）、ファイルの大きさ（File Size）、管理するファイルが含まれるディレクトリのDirectory Descriptorの記録位置を示す論理ブロック番号（Pointer toParent Directory）、ファイル構造でディスク上の連続領域を管理する場合の位置情報である開始論理ブロック番号と論理ブロック数（Location of Contiguous Extent）、このファイルを参照している仮想ファイルの数（Number of References by Virtual File）、このファイルを参照している仮想ファイルを管理するVirtual File Descriptorの記録位置である論理ブロック番号（Location of Virtual File Descriptor）、ファイルが管理するExtent Spaceに記録されるデータのディスク上での分断数（Number of Extents）、そしてそれぞれの分断の位置情報である開始論理ブロック番号と論理ブロック数（Location of Extent）で構成される。なお、Location of Virtual File DescriptorはNumber of References by Virtual Fileで管理される参照数の数だけ記録され、同様にLocation ofExtentはNumber of Extentsで管理される分断数の数だけ記録される。

【００９５】図21にFile Descriptorの属性情報であるAttributeについて示す。Attributeは16bitの情報であり、Bit0 Read Onlyは管理するファイルが読み込み専用であるかを示し、Bit1 Deletedは管理するファイルが一時的に削除されたかどうかを示し、Bit2 Contiguousは管理するファイルがディスク上の連続領域を確保しているかどうかを示し、Bit3 Allocation Modeはファイルで管理されるデータの配置方法がAreaの前方からアクセスであるか後方からのアクセスであるかを示しBit4、5 CGMS（Copy Generation Management System）は、コピーを許可するか、1世代のみコピーするか、コピーを禁止するかどうかの情報を示す。なおBit6-15までは将来の拡張用にReservedされたビットである。

【００９６】ここで、オリジナルシーンなどのファイルを作成する際のデバイスドライバの処理手順を示すフローチャートを図22に示す。ユーザによって映像が撮影されステップS10においてオリジナルシーンのファイル書き込み要求がデバイスドライバに対して発生すると、ステップS11においてまずオリジナルシーンを管理するFile Descriptorが記録されるArea Management Spaceに空き論理ブロックがあるかどうかと、書き込むユーザデータの大きさの空き領域がExtent SpaceにあるかどうかをSpace Management DescriptorのSpace bitmapを利用することによって把握し記録位置を確定する。

【００９７】ステップS12において、もしExtent Spaceに空き領域が無いようであれば、ステップS18においてエラー処理を行ない終了する。ステップS12においてExtent Spaceに空き領域があるようであれば、ステップS13において実際の映像データをExtent Spaceに書き込む。

【００９８】ステップS14において、記録する全てのデータが書き込まれたかどうかを判断する。一般的に映像データは巨大であるため、撮影したオリジナルシーンのデータを一度に記録することはできないので、あるデータ量単位でデータの記録をしていく。もしまだ記録するデータが残っているようであれば、ステップS11に戻って処理を繰り返す。

【００９９】ステップS14において記録する全てのデータがExtent Spaceに記録されたと判断された場合は、ステップS15において記録したオリジナルシーンのデータのExtent Spaceでの位置を管理するFile DescriptorをArea Management Spaceに書き込む。

【０１００】ステップS16において、作成したファイルが含まれるディレクトリを管理するDirectory Descriptorに、作成したファイルへのDescriptor Pointer Entryを追加する。ステップS17において、Extent Spaceに記録したオリジナルシーンのデータと、作成したファイルを管理するFile Descriptorを記録したArea Management Spaceに対応するSpace Management DescriptorのSpace Bitmapを使用状態に更新し、ファイル作成処理を終了する。このようにして、ユーザが撮影開始から撮影停止までの１シーンをオリジナルシーンとして記録媒体上に記録することができる。

【０１０１】図23に示すように、記録開始から停止あるいは一時停止に対応するオリジナルシーンに対して編集を行なうことがある。前述したように本発明では、非破壊編集を基本としている。非破壊編集とは、ディスク上のExtent Spaceに記録されたオリジナルシーンのデータをコピーしたり移動する事無く、オリジナルシーンの任意の箇所を選択してユーザシーンを定義し、それらを任意の順番で組み合わせる事によってユーザプログラムを定義することを意味する。

【０１０２】ユーザシーンを定義するには、まずユーザがオリジナルシーンの任意の箇所を選択する事から始まる。例えば、任意の箇所を選択するには、編集アプリケーションプログラムを用いて映像の選択箇所の開始点と終了点を指定することになる。この再生開始と終了点を指定した後、選択箇所の映像に対応するディスク上の映像データの記録位置を管理する仮想ファイルを作成する。

【０１０３】この時、仮想ファイル名をユーザシーンの再生順番に従ってUS0001.MPG、US0002.MPG、US0003.MPGと言ったように、数字の部分で順番が把握できるようにする。ユーザシーンを再生順序に従って組み合わせた管理単位をユーザプログラムと呼び、1つの方法として連続領域を確保しない通常のディレクトリとして管理する。ディレクトリ名は、UPRG0001のように定義し、数字の部分はユーザプログラム番号に相当し、ユーザプログラムを新規に作成する度にUPRG0001、UPRG0002、UPRG0003と言ったように作成されていく。ここで、ユーザシーンを管理する仮想ファイルやユーザプログラムを管理するディレクトリは、あくまでも論理ファイルシステムの管理情報であるので、デバイスドライバが新規に作成する場合はディスク上のArea Management Spaceのみにアクセスが発生する。つまりユーザデータ領域でExtent Spaceに対する書き込みや変更は一切発生しない。

【０１０４】ユーザシーンを管理する仮想ファイルの論理ファイルシステムの管理情報であるVirtual File Descriptorの内容を図24に示す。Virtual File Descriptor（VFD）はVFDを識別するためのID（Header ID）、１論理ブロックにVFDが格納できずに複数の論理ブロックにまたがって記録される場合に次にアクセスすべき論理ブロック番号（Next Extension）、同様に複数論理ブロックにまたがった場合に1つ前の情報が記録された論理ブロック番号（Previous Extension）、ファイルの属性情報（Attribute）、ファイル名（File Name）、ファイルの作成および修正時刻（Creation Time & Date、 Modified Time & Date）、ファイルの大きさ（File Size）、管理するファイルが含まれるディレクトリのDirectory Descriptorの記録位置の論理ブロック番号（Pointer to Parent Directory）、この仮想ファイルが参照しているファイルを管理するFile Descriptorの記録位置の論理ブロック番号（Location of Referenced File Descriptor）、仮想ファイルが指し示す選択箇所のディスク上の分断数（Number of Extents）、そしてそれぞれの分断の位置情報である開始論理ブロック番号と論理ブロック数（Location of Extent）で構成される。このとき、Location of ExtentはNumber of Extentsで管理される値の数だけ存在することになる。

【０１０５】図25にVirtual File Descriptorの属性情報であるAttributeについて示す。Attributeは16bitの情報であり、Bit0 Read Onlyは管理するファイルが読み込み専用であるかを示し、Bit1 Deletedは管理するファイルが一時的に削除されたかどうかを示し、Bit2 Conti

guousおよびBit3 Allocation Modeは使用しない、Bit 4、5 CGMS (Copy Generation Management System) は、コピーを許可するか、1世代のみコピーするか、コピーを禁止するかどうかの情報を示す。なおBit6-15までは将来の拡張用にReservedされたビットである。

【０１０６】ここで、図26に仮想ファイルを作成する際の処理手順を説明するフローチャートを示す。デバイスドライバに対して、ユーザシーンを管理する仮想ファイルの作成要求がステップS20において発生すると、ステップS21においてSpace Management DescriptorのSpace bitmapの情報からArea Management Spaceに空き論理ブロックがあるかどうかを確認する。ステップS22においてもし空きスペースが無い場合はステップS30においてエラー処理をして処理を終了する。

【０１０７】ステップS22においてArea Management Spaceに空き論理ブロックがある場合はステップS23において作成する仮想ファイルが参照するオリジナルシーンのFileDescriptorをArea Management Spaceから読み出す。

【０１０８】ステップS24において作成する仮想ファイルの管理情報であるVirtual File DescriptorをArea Management Spaceに書き込む。このときこの仮想ファイルはユーザが指定したユーザシーンの対応するディスク上の位置情報Location of Extentsの集合を保持していることになる。

【０１０９】ステップS25において、ステップS23において読み込んだオリジナルシーンの管理情報であるFile Descriptor内のオリジナルシーンを参照している仮想ファイル数を示すNumber of References by Virtual Fileの値に1を足す。

【０１１０】ステップS26において同様に、File Descriptor内のオリジナルシーンを参照している仮想ファイルを管理するVirtual File DescriptorのArea Management Space内の位置情報であるLocation of Virtual File Descriptorに、ステップS24においてディスクに書き込んだVirtual File Descriptorのディスク上での記録位置である論理ブロック番号を追加する。

【０１１１】ステップS27において、ステップS25およびステップS26において更新した参照しているオリジナルシーンのFile Descriptorをディスク上で更新をする。

【０１１２】ステップS28において、ステップS24において作成した仮想ファイルが含まれるディレクトリを管理するDirectory Descriptorに作成したVirtual File DescriptorへのポインタであるDescriptor Pointer Entryを追加する。

【０１１３】ステップS29において、ステップS24において新規に作成したVirtual File DescriptorのArea Management Space内の記録位置に対応するSpace Management DescriptorのSpace Bitmapを更新し処理を終了する。

【０１１４】以上のような処理手順によって、仮想ファイルを定義することができ、デバイスドライバを介してディスクにアクセスするユーザシステムにとって、オリジナルシーンを管理する実ファイルおよび、ユーザシーンを管理する仮想ファイルを、対応するディスク上のデータとして記録することが可能となる。つまりは、実ファイルの管理情報、仮想ファイルの管理情報を記録することが可能となる。

【０１１５】オリジナルプログラムやユーザプログラムを再生するには、目的のプログラムのディレクトリ名を指定し、そのディレクトリの下に保存されているオリジナルシーンやユーザシーンの実ファイルあるいは仮想ファイルを順番に再生していく事で目的が達成できる。

【０１１６】図２３に記録媒体の様子を示す。Rootディレクトリの下に、ディレクトリORIGINALが作成されており、その中に実ファイル（オリジナルシーン）OS0001.MPG,OS0002.MPG,OS0003.MPGが記録されている。このそれぞれにFile Descriptorを備えている。

【０１１７】各実ファイルに、それぞれ仮想ファイルが定義されている。OS0001.MPGには仮想ファイルUS0001.MPGが、OS0002.MPGには仮想ファイルUS0003.MPGが、OS0003.MPGには仮想ファイルUS0002.MPGが定義されており、それぞれがVirtual File Descriptorを備えている。この３つの仮想ファイルは、ディレクトリUPRG0001で管理されているため、再生時にディレクトリUPRG0001が指定されると、US0001.MPG,US0002.MPG,US0003.MPGの順に記録媒体上から読み出して再生することが可能である。

【０１１８】例えば作成したユーザプログラムを他のディスクにコピーして他の人にあげたり、IEEE1394などを用いてネットワーク経由でPCやその他のAV機器に転送することも考えられる。オリジナルシーンやユーザシーンのコピーやネットワーク転送を行なうには、それぞれ対応するファイルや仮想ファイルを指定してファイルコピーを行なったり、ファイル転送を行なえば良い。またオリジナルプログラムやユーザプログラムをコピーやネットワーク転送するには、それぞれ対応するディレクトリを指定して、そのディレクトリ以下のファイルのコピーや転送を行なえば良い。

【０１１９】次にユーザシーンを削除する場合の手順について説明する。まずユーザシステムによってユーザシーンの削除要求が発生した場合、このユーザシーンを管理している仮想ファイルの論理ファイルシステムの管理情報である、Virtual File DescriptorをArea Management Spaceから削除する。仮想ファイルはあくまでもオリジナルシーンで管理されているオリジナルのデータへの参照情報に過ぎないため、Extent Spaceに記録されている参照している映像データ（実データ）を削除することは許されない。図27に従って本実施形態での仮想ファイルの削除手順について説明する。

【０１２０】ステップS40において、デバイスドライバに対してユーザシーンを管理する仮想ファイルの削除要

求が発生した場合、ステップS41において対象の仮想ファイルの管理情報であるVirtual File DescriptorをArea Management Spaceから読み出す。

【０１２１】ステップS42において、参照していたオリジナルシーンの管理情報であるFileDescriptorをArea Management Spaceから読み出す。この際、ステップS41で読み出したVirtual File Descriptor内のLocation of Referenced File Descriptorに記録されている論理ブロック番号にアクセスする。

【０１２２】ステップS43において、削除する仮想ファイルが含まれていたディレクトリを管理するDirectory Descriptorをディスクから読み出し、削除する仮想ファイルへのポインタ情報であるDescriptor Pointer Entryを削除し、管理情報をディスク上で更新する。

【０１２３】ステップS44で、ステップS42において読み込んだオリジナルシーンのFile Descriptor内のNumber of References by Virtual Fileの値から1を引く。ステップS45において同様に、File Descriptor内のLocation of Virtual File Descriptorから削除したVirtual File Descriptorの論理ブロック番号を削除する。ステップS46においてステップS44、S45において更新したFile Descriptorをディスク上で更新する。

【０１２４】ステップS47において削除したVirtual File Descriptorが記録されていたAreaManagement Spaceに対応するSpace Management DescriptorのSpace Bitmapを解放し処理を終了する。

【０１２５】つぎに、オリジナルシーンの削除処理について説明する。既に述べたようにユーザプログラムはユーザシーンの集合であり、ユーザシーンは仮想ファイルとしてオリジナルシーンの実ファイルを参照している。ユーザの視点で見ると、ディスク上でのオリジナルシーンのデータをユーザシーンが参照しているだけであっても、各ユーザシーンやユーザプログラムはそれぞれオリジナルシーンやオリジナルプログラムとは異なる目的や用途によって作成されたものである。オリジナルシーンの実ファイルを単純に消去してしまうと問題が起きる。これは、オリジナルシーンのデータを消去してしまうことによって、ユーザシーンである仮想ファイルの参照先であるディスク上のデータにアクセスできなくなってしまうからである。よって、オリジナルシーンのデータを完全に削除したり、部分に削除したりする前には、削除しようとするオリジナルデータを参照している仮想ファイルがあるかどうかを調べる必要がある。このようにどの仮想ファイルから参照されているかどうかを管理することによって、仮想ファイルから参照されている場合オリジナルシーンの削除を抑制したり、仮想ファイルが参照していない部分だけを削除したり、参照している仮想ファイルも一緒に削除するといった様々な制御を行うことが可能となる。

【０１２６】図28に従って本実施形態におけるオリジナルシーンを管理するファイルの削除処理について説明していく。ステップS50においてデバイスドライバに対してオリジナルシーンの削除要求が発生すると、ステップS51において、削除しようとするオリジナルシーンを管理しているFile DescriptorをArea Management Spaceから読み出す。

【０１２７】ステップS52において、読み込んだFile Descriptor内のNumber of Referencesby Virtual Fileの内容より仮想ファイルから参照されているかどうかを判断する。ステップS52において、仮想ファイルから参照されていない、つまり参照数が0の場合、ステップS57において、オリジナルシーンを管理するファイルが含まれていたディレクトリを管理するDirectory Descriptorから削除するFile DescriptorへのポインタであるDescriptor Pointer Entryを削除しディスク上で更新する。

【０１２８】ステップS58において、削除対象のArea Management Spaceに記録されたFile DescriptorとExtent Spaceに記録されたユーザデータに対応するSpace Management DescriptorのSpace bitmapを更新し処理を終了する。

【０１２９】ステップS52において、削除しようとするオリジナルシーンが仮想ファイルから参照を受けている場合は、ステップS53において、消去しようとしているオリジナルシーンの仮想ファイルによって参照されていない部分のみ削除するか、参照している仮想ファイルも一緒に削除するか、あるいは削除を中止するかを決める。例えば、ユーザシステムにおいて、ユーザに消去しようとしているオリジナルシーンがユーザシーンから参照されていることを警告し、処理をユーザに選ばせることができる。ステップS53において、削除処理を中止するのであれば、処理を終了する。

【０１３０】ステップS53において、オリジナルシーンを消去するとともに仮想ファイルも一緒に削除する処理が選ばれた場合、ステップS59において、読み込まれた削除しようとしているFile Descriptor内のLocation of Virtual File Descriptorの情報から参照している仮想ファイルを管理する全てのVirtual File DescriptorのArea Management Space内での記録位置を把握し、全てのVirtual File Descriptorを読み出す。

【０１３１】ステップS60において、読み込んだVirtual File Descriptor内のPointer to Parent Directoryによって削除しようとする仮想ファイルが含まれるディレクトリを管理するDirectory Descriptorを削除しようとする仮想ファイル分読み出す。

【０１３２】ステップS61において、読み込んだDirectory Descriptorから削除する仮想ファイルへのポインタであるDescriptor Pointer Entryを削除して、Directory Descriptorをディスク上で更新する。

【０１３３】ステップS57において、削除するオリジナルファイルが含まれるディレクトリを管理するDirector

y Descriptorから削除するFile Descriptorへのポインタ情報であるDescriptor Pointer Entryを削除し、Directory Descriptorをディスク上で更新する。

【０１３４】ステップS58において、削除したオリジナルシーンの論理ファイルシステムの管理情報であるFile Descriptorと、Extent Space内の削除する映像データと、オリジナルシーンを参照していた仮想ファイルの論理ファイルシステムの管理情報であるVirtual File Descriptorの記録位置に対応するSpace Management DescriptorのSpace bitmapを開放し処理を終了する。

【０１３５】一方、ステップS53において、仮想ファイルから参照されていない部分のみ削除する場合は、ステップS54において、削除しようとしたオリジナルシーンを参照している仮想ファイルを管理するVirtual File Descriptorをすべて Area Management Spaceより読み出す。この際、ステップS51において読み出したFile Descriptor内のNumber of References by Virtual File DescriptorとLocation of Virtual File Descriptorを利用する。

【０１３６】ステップS55において、読み出した全ての仮想ファイルの管理情報であるVirtual File Descriptor内のNumber of ExtentsとLocation of Extentを利用することによって、オリジナルシーンのどの部分が仮想ファイルから参照を受けていないかを把握する。

【０１３７】ステップS56において、オリジナルシーンを管理するFile DescriptorのNumberof ExtentsおよびLocation of Extentで、仮想ファイルが参照している全ての参照箇所を含めるように、オリジナルシーンが管理するExtent Spaceの位置情報を更新する。

【０１３８】ステップS58において、オリジナルシーンの内、仮想ファイルから参照を受けていなかった部分に対応するSpace Management DescriptorのSpace bitmapを更新し、処理を終了する。

【０１３９】次に、ファイルを管理するFile Descriptorと仮想ファイルを管理するVirtualFile Descriptor、仮想ファイルを任意に組み合わせて新たなファイル構造により管理を行う仮想プログラムファイルVirtual Program File Descriptorを有する第2の実施形態に関して説明する。

【０１４０】図29に第２の実施形態の概要を示す。この図では、ファイル名をユーザシーンの作成順番に従ってUS0001.MPG、US0002.MPG、US0003.MPGと言ったように、数字の部分で記録順番が把握できるようにする。ユーザシーンを作成順序に従って組み合わせた管理単位をユーザプログラムと呼び、1つの方法として連続領域を確保しない通常のディレクトリとして管理する。ディレクトリ名は、UPRG0001のように定義し、数字の部分はユーザプログラム番号に相当し、ユーザプログラムを新規に作成する度にUPRG0001、UPRG0002、UPRG0003と言ったように作成されていく。

【０１４１】ここで、ユーザシーンを管理する仮想ファイルやユーザプログラムを管理するディレクトリは、あくまでも論理ファイルシステムの管理情報であるので、デバイスドライバが新規に作成する場合はディスク上のArea Management Spaceのみにアクセスが発生する。つまりユーザデータ領域でExtent Spaceに対する書き込みや変更は一切発生しない。

【０１４２】オリジナルプログラムやユーザプログラム全体を１つの仮想プログラムファイルとして扱うことも可能である。オリジナルプログラムを管理する仮想プログラムファイルをOP0000.PRGとし、ユーザプロラムを管理する仮想プログラムファイルをUP0001.PRGとする。1つのプログラムを管理する仮想プログラムファイルは、定義されたプログラムを構成するオリジナルシーンやユーザシーンを管理する仮想ファイルへのポインタ情報で構成されている。よって仮想プログラムファイルによって、一連の読み込むべきオリジナルシーンやユーザシーンの管理情報を把握するとともに、コピーなどの編集を容易に行うことができる。

【０１４３】つまり、UP0001.PRGという仮想プログラムファイルをコピーするということは、ユーザプログラム1を構成する全てユーザシーンのデータを指定してコピーすること等しいことを意味する。ユーザプログラムを追加する場合は、新たにユーザプログラム用のディレクトリを作成し、その中にユーザシーンを管理する仮想ファイルと、ユーザプログラムを管理する仮想プログラムファイルを作成していく事になる。

【０１４４】オリジナルプログラムやユーザプログラムを再生するには、目的のプログラムのディレクトリ名を指定し、そのディレクトリの下に保存されているオリジナルシーンやユーザシーンのファイルを順番に再生していく事で目的が達成できる。また、オリジナルプログラムやユーザプログラムを管理する仮想プログラムファイルが定義されている場合は、その仮想プログラムファイルを指定することによって、読み込むべきデータを管理している管理情報を把握することが可能となる。

【０１４５】例えば作成したユーザプログラムを他のディスクにコピーして他の人にあげたり、IEEE1394などを用いてネットワーク経由でPCやその他のAV機器に転送することも考えられる。オリジナルシーンやユーザシーンのコピーやネットワーク転送を行なうには、それぞれ対応するファイルを指定してファイルコピーを行なったり、ファイル転送を行なえば良い。

【０１４６】またオリジナルプログラムやユーザプログラムをコピーやネットワーク転送するには、それぞれ対応するディレクトリを指定して、そのディレクトリのコピーや転送を行なえば良い。また、仮想プログラムファイルの機能を利用すれば、プログラムに関してもファイルで管理できるので、オリジナルプログラム、ユーザプログラムに関しても、それぞれ対応する仮想プログラム

ファイルを指定することによって、ファイル単位のコピーを行なったり、ファイル単位の転送を行なう事が可能となる。

【０１４７】仮想プログラムファイルを定義する機能は、説明したユーザプログラムを表現するためだけではなく異なった使い方もできる。

【０１４８】例えば、ある規格において定義されるストリームデータがあるとする。このストリームデータはそのストリームの内容を管理するヘッダ情報とデータ情報とで構成されるものとする。ここでこの情報をディスクに記録する際に、そのストリームデータを読むのに重要なデータであるヘッダ情報をディスク上の信頼できる領域に置いて、データは通常の領域に置くことも考えられる。

【０１４９】図30に示す例では、連続領域を確保するディレクトリManagementとディレクトリDataが定義されている。Managementは管理情報を保存するためのディレクトリであり、ディスク上ではサーティファイ処理などを行なっておりより信頼度の高い領域を使用している。

【０１５０】一方Dataはデータを保存するためのディレクトリであり信頼度の関してはManagement領域と比較して落ちる領域である。このディスクを使用するアプリケーションプログラムは、ヘッダ情報をHEADER.INFOというファイルでManagementディレクトリの下で管理を行なっている。また実データはDataディレクトリの下でDATA.DATとして記録されている。

【０１５１】管理情報であるHEADER.INFOが実データと分割して記録されていることによって、管理情報の読み込みや更新が楽に行なえ、信頼度の高い領域に書き込むことが可能となる。そして、前述した仮想プログラムファイル機能を用いて、HEADER.INFOとDATA.DATを連結して仮想的に1つのファイルとして見せるSTREAM.DATという仮想プログラムファイルをManagementディレクトリの下に記録する。

【０１５２】このディスクを使用するアプリケーションのみで使うと言った閉じた世界では全く問題にならないが、2つのファイルに分割して記録されているデータは、それぞれ規格に合致しない内容のものである。しかし、仮想プログラムファイルであるSTREAM.DATにアクセスすることによって、実際には2つのファイルとして分割して管理されているストリームデータであっても、1つのストリームデータとしてアクセスすることができ、互換性を保つことも可能となる。この事により、規格に準拠したデータを扱うアプリケーションプログラムさえあれば、仮想プログラムファイルを通してアクセスすることによってデータの読み込みが行えることになる。

【０１５３】実施形態2におけるオリジナルシーンを管理するFile Descriptorの構造および、ファイルの作成手順は実施形態1と同じため説明を省く。図31に実施形態2における仮想ファイル、Virtual File Descriptor (VFD) の説明を示す。

【０１５４】Virtual File Descriptor (VFD)はVFDを識別するためのID (Header ID) 、１論理ブロックにVFDが格納できずに複数の論理ブロックにまたがって記録される場合に、次にアクセスすべき論理ブロック番号 (Next Extension) 、同様に複数論理ブロックにまたがった場合に、１つ前の情報が記録された論理ブロック番号 (Previous Extension) 、ファイルの属性情報 (Attribute) 、ファイル名 (File Name) 、 VFDの作成および修正時刻 (Creation Time & Date、Modified Time& Date) 、ファイルの大きさ (File Size) 、管理するファイルが含まれるディレクトリのDirectory Descriptorの記録位置である論理ブロック番号 (Pointerto Parent Directory) 、この仮想ファイルが参照しているファイルを管理するFile Descriptorの記録位置である論理ブロック番号 (Location of Referenced File Descriptor) 、この仮想ファイルを参照している仮想プログラムファイルの数 (Number of References by Virtual Program) 、この仮想ファイルを参照している仮想プログラムファイルを管理するVirtual Program File Descriptorの記録位置である論理ブロック番号 (Location of Virtual Program File Descriptor) 、仮想ファイルが指し示す選択箇所のディスク上の分断数 (Number of Extents) 、そしてそれぞれの分断の位置情報である開始論理ブロック番号と論理ブロック数 (Location of Extent) で構成される。このとき、Location of Virtual Program File DescriptorはNumber of References by Virtual Programの数だけ存在し、Location of ExtentはNumber of Extentsで管理される値の数だけ存在することになる。

【０１５５】図25にVirtual File Descriptorの属性情報であるAttributeについて示す。実施形態1において説明した場合と同じなので説明は省く。図32に実施形態2における仮想プログラムファイル、Virtual Program File Descriptor (VPFD) の説明をする。 VPFDはVPFDを識別するためのID (Header ID) 、１論理ブロックにVPFDが格納できずに複数の論理ブロックにまたがって記録される場合に次にアクセスすべき論理ブロック番号 (Next Extension) 、同様に複数論理ブロックにまたがった場合に１つ前の情報が記録された論理ブロック番号 (Previous Extension) 、ファイルの属性情報 (Attribute) 、ファイル名 (FileName) 、 VPFDの作成および修正時刻 (Creation Time & Date、 Modified Time& Date) 、ファイルの大きさ (File Size) 、管理するファイルが含まれるディレクトリのDirectory Descriptorが記録されている論理ブロック番号 (Pointerto Parent Directory) 、このファイルを構成するための仮想ファイルの数 (Number of Referenced Virtual Files) 、そしてそれぞれの仮想ファイルを管理するVirtual File Descriptorが記録されているディスク上での位置情報で論理

ブロック番号（Location of Virtual File Descriptor）で構成される。このとき、Location of Virtual File DescriptorはNumber of Referenced Virtual Filesで管理される値の数だけ存在することになる。

【０１５６】Virtual Program File Descriptorによって、全てのオリジナルシーンを組み合わせたオリジナルプログラムも管理することが可能である。この場合は、Number of Referenced Virtual Filesには全てのオリジナルシーンの数、Location of Virtual File Descriptorには、オリジナルシーンを管理するFile Descriptorの記録位置を書き込む。

【０１５７】Virtual Program File Descriptorの属性情報は、図25で示すVirtual File Descriptor用のAttributeと同一である。また、実施形態２において、仮想ファイル作成の処理手順は、図26に示すように、実施形態１と同様であるため、説明は省略する。

【０１５８】次に、実施形態２における、仮想プログラムファイルを作成する際のデバイスドライバの処理手順を説明する。図33は処理手順を示すフローチャートである。オリジナルプログラムやユーザプログラムを管理する仮想プログラムファイルの作成要求がステップS70において発生すると、ステップS71においてSpace Management DescriptorのSpace bitmapの情報から管理領域であるArea Management Spaceに空き論理ブロックがあるかどうかを確認する。

【０１５９】ステップS72において、もし空きスペースが無いと判断された場合は、ステップS82において、エラー処理をして処理を終了する。ステップS72において、空きスペースがあると判断された場合はステップS73において、作成する仮想プログラムファイルがオリジナルプログラムかユーザプログラムを管理するものかを判断する。もしステップS73においてユーザプログラムと判断された場合、ステップS74において、作成する仮想プログラムファイルが参照する全てのユーザシーンのVirtual File Descriptorをディスクから読み出す。

【０１６０】ステップS75において、新規に作成する仮想プログラムファイルの管理情報である、Virtual Program File DescriptorをArea Management Spaceに記録する。このとき、この仮想プログラムファイルは、ユーザが指定したユーザシーンを管理する仮想ファイルの管理情報であるVirtual File Descriptorの数と、それぞれの記録位置を保持していることになる。

【０１６１】ステップS76において、ステップS74において読み込んだユーザシーンの管理情報であるVirtual File Descriptor内のユーザシーンを参照している仮想プログラムファイル数を示すNumber of References by Virtual Programの値に1を足す。

【０１６２】ステップS47において同様に、Virtual File Descriptor内のユーザシーンを参照している仮想プログラムファイルの記述子の位置情報であるLocation of Virtual Program File Descriptorに、ステップS75においてディスクに書き込んだVirtual Program File Descriptorのディスク上での記録位置である論理ブロック番号を保持する。

【０１６３】ステップS78において、ステップS76およびステップS77において更新した参照しているユーザシーンのVirtual File Descriptorをディスク上に書き直して更新する。ステップS79において、ステップS45において作成した仮想プログラムファイルが含まれるディレクトリを管理するDirectory Descriptorに作成したVirtual Program File DescriptorへのポインタであるDescriptor Pointer Entryを追加する。

【０１６４】ステップS80において、ステップS75において新規に作成したVirtual ProgramFile Descriptorに対応するSpace Management Descriptorに対応するSpace Bitmapを更新し処理を終了する。

【０１６５】もしステップS73においてオリジナルプログラムと判断されたら、ステップS81において新規に作成する仮想プログラムファイルの管理情報である、Virtual Program File DescriptorをArea Management Spaceに記録する。このときこの仮想プログラムファイルはオリジナルシーンを管理するファイルの管理情報であるFile Descriptorの数と記録位置を保持していることになる。

【０１６６】ステップS79において、ステップS81において作成した仮想プログラムファイルが含まれるディレクトリを管理するDirectory Descriptorに、作成したVirtualProgram File DescriptorへのポインタであるDescriptor Pointer Entryを追加する。ステップS80において、ステップS81において新規に作成したVirtual Program File Descriptorに、対応するSpace Management Descriptorに対応するSpace Bitmapを更新し処理を終了する。

【０１６７】ここで、ユーザプログラムの場合と異なり、オリジナルプログラムを管理する仮想プログラムファイルを作成する際に、参照されているオリジナルシーンの管理情報であるFile Descriptorに参照情報を記録しない理由は、オリジナルプログラムの性質からくるものである。オリジナルプログラムは１つのディスクに１つしか存在せず、また全てのオリジナルシーンがオリジナルプログラムに含まれないといけないので、参照関係をあえて管理する必要がないことになる。

【０１６８】次に、ユーザシーンやユーザプログラムを削除する場合の手順について説明する。まずユーザによってユーザシーンやユーザプログラムの削除要求が発生した場合、このユーザシーンやユーザプログラムを管理している論理ファイルシステムの管理情報をディスクから削除する。仮想ファイルはあくまでもオリジナルシーンで管理されているオリジナルのデータへの参照情報に過ぎないため、実際のディスクの領域（実ファイルの領

域）を解放することは許されない。

【０１６９】ここで図34に実施形態２における、仮想ファイルを削除する際のデバイスドライバの処理手順を説明するフローチャートを示す。ステップS90において、ユーザシーンを管理する仮想ファイルの削除要求が発生した場合、ステップS91において、削除対象の仮想ファイルの論理ファイルシステムの管理情報であるVirtual File DescriptorをArea Management Spaceから読み出す。

【０１７０】ステップS92においてステップS91において読み込んだ論理ファイルシステムの管理情報を元に、当該仮想ファイルが、仮想プログラムファイルから参照されているかどうかを把握し判断する。ステップS92において削除しようとする仮想ファイルが仮想プログラムファイルから参照されていない場合は、ステップS93において、参照していたオリジナルシーンの論理ファイルシステムの管理情報であるFile DescriptorをArea Management Spaceから読み出す。

【０１７１】ステップS94において削除する仮想ファイルが含まれていたディレクトリを管理するDirectory DescriptorをArea Management Spaceから読み出し、削除する仮想ファイルを示すDescriptor Pointer Entryを削除しディスク上で更新する。

【０１７２】ステップS95で、ステップS93において読み込んだオリジナルシーンを管理するFile Descriptor内のNumber of References by Virtual Fileの値から1を引く。ステップS96において同様に、File Descriptor内のLocation of Virtual File Descriptorから、削除したVirtual File Descriptorのアドレスを削除する。ステップS97においてステップS95、S96において更新したFile Descriptorをディスク上で更新する。ステップS98において削除したVirtual File Descriptorが記録されていた領域に対応するSpace Management DescriptorのSpace Bitmapを解放し、処理を終了する。

【０１７３】ステップS92において、削除しようとする仮想ファイルが仮想プログラムファイルから参照されている場合は、ステップS99において、仮想プログラムからユーザシーンを削除するかを判断する。ステップS99において削除しないと判断した場合は、ステップS103においてエラー処理をし処理を終了する。

【０１７４】ステップS99においてユーザシーンを削除すると判断した場合、ステップS100において参照されていた仮想プログラムファイルの管理情報であるVirtual Program File DescriptorをArea Management Spaceから読み出し、参照先のユーザシーンを管理するNumber of Referenced Virtual Fileの値から1引く。

【０１７５】ステップS101において同様にVirtual Program File DescriptorのLocation ofVirtual File Descriptorから、削除するVirtual File Descriptorのアドレス情報を削除する。ステップS102において、ステップS100、S101において更新した仮想プログラムファイルの管理情報をディスク上で更新し、ステップS93に移る。後は処理は前述の通りである。

【０１７６】次に、実施形態２における、仮想プログラムファイルを削除する際のデバイスドライバの処理手順を説明する。図35はそのフローチャートである。ステップS110において、仮想プログラムファイルの削除要求が発生した場合、ステップS111において対象となる仮想プログラムファイルの管理情報であるVirtual ProgramFile DescriptorをArea Management Spaceから読み出す。

【０１７７】ステップS112において、削除する仮想プログラムファイルがユーザプログラムかオリジナルプログラムかどうかを判断する。ステップS112において、ユーザプログラムと判断された場合、ステップS113において、仮想プログラムファイルが参照していた全ての仮想ファイルの管理情報であるVirtual File Descriptorを、Area Management Spaceから読み出す。

【０１７８】ステップS114において読み出した全てのVirtual File Descriptor内のNumberof References by Virtual Programの値から1引く。ステップS115において同様にVirtual File Descriptor内のLocation of Virtual Program File Descriptorから、削除する仮想プログラムファイルのアドレス情報を削除する。ステップS116において、ステップS114、S115において更新したVirtual File Descriptorを全てディスク上で更新する。

【０１７９】ステップS117において、削除した仮想プログラムファイルが含まれていたディレクトリを管理するDirectory Descriptorから、削除した仮想プログラムファイルへのポインタであるDescriptor Pointer Entryを削除する。

【０１８０】ステップS118において、削除したVirtual Program File Descriptorが使用していたArea Management Space内の論理ブロックに対応するSpace Management DescriptorのSpace bitmapを解放して処理を終了する。

【０１８１】ステップS112において、オリジナルプログラムと判断された場合、ステップS117において削除した仮想プログラムファイルが含まれていたディレクトリを管理するDirectory Descriptorから、削除した仮想プログラムファイルへのポインタであるDescriptor Pointer Entryを削除する。

【０１８２】ステップS118において削除したVirtual Program File Descriptorが使用していたArea Management Space内の論理ブロックに対応するSpace Management DescriptorのSpace bitmapを解放して処理を終了する。この際、ステップS112においてオリジナルプログラムと判断された場合、オリジナルプログラムを消去してはならないという仕様であれば、エラー処理を行って処理を終了するといった処理を取ってもよい。

【０１８３】次に、オリジナルシーン（実ファイル）に

対して削除要求が行われた場合について説明する。上記したような処理では、オリジナルシーンを参照しているユーザシーン（仮想ファイル）が他に存在する場合は、基本的にはオリジナルシーンは削除しない。これは、オリジナルシーンを参照しているユーザシーンが他にある場合、オリジナルシーンを削除してしまうと、参照先のデータが削除されたことになり、そのオリジナルシーンを参照している全てのユーザシーンが再生不可能となってしまうからである。

【０１８４】しかしながら、参照しているユーザシーンが、オリジナルシーンの一部しか参照していない場合、オリジナルシーンを再生しない限り、全く再生されないデータを多く記録媒体に記録していることになり、記録の効率としては好ましくない。

【０１８５】そこで、オリジナルシーンの削除要求があった場合には、オリジナルデータのうち、どのユーザシーンにも参照されていない部分のオリジナルデータを削除することによって、記録媒体の領域を有効に使用することができる。このような処理を行うためには、オリジナルシーンのデータを完全に削除したり、部分に削除したりする前に、削除しようとするオリジナルシーンのデータを参照している仮想ファイルがあるかどうか、またオリジナルシーンのどの部分が参照されているかを調べる必要がある。

【０１８６】ここで図36に実施形態２において、オリジナルシーンを管理するファイルを削除する際のデバイスドライバの処理手順を説明するフローチャートを示す。ステップS120において、デバイスドライバに対してオリジナルシーンの削除要求が発生すると、ステップS121において、削除しようとするオリジナルシーンを管理しているFile DescriptorをArea Management Spaceから読み出す。

【０１８７】ステップS122において、読み込んだFile Descriptor内のNumber of References by Virtual Fileの内容より仮想ファイルから参照されているかどうかを判断する。ステップS122において仮想ファイルから参照されていない、つまり参照数が0の場合、このオリジナルシーンは削除しても影響がないので、ステップS127においてオリジナルシーンを管理するファイルが含まれていたディレクトリを管理するDirectory Descriptorから削除するFile DescriptorへのDescriptor Pointer Entryを削除する。

【０１８８】ステップS128において、削除対象のArea Management Spaceに記録されたFileDescriptorとExtent Spaceに記録されたユーザデータに対応するSpace Management DescriptorのSpace bitmapを更新し処理を終了する。

【０１８９】ステップS122において削除しようとするオリジナルシーンが仮想ファイルから参照を受けている場合は、ステップS123において、消去しようとしているオリジナルシーンが仮想ファイルによって参照されていない部分のみ削除するか、参照している仮想ファイルも一緒に削除するか、あるいは削除を中止するかを決める。例えば、ユーザシステムにおいて、ユーザに消去しようとしているオリジナルシーンがユーザシーンから参照されていることを警告し、処理をユーザに選ばせることができる。

【０１９０】ステップS123において、削除処理を中止するのであれば、処理を終了する。ステップS123において、オリジナルシーンを消去するとともに仮想ファイルも一緒に削除する処理が選ばれた場合、ステップS129において読み込まれた、削除しようとしているFile Descriptor内のLocation of Virtual File Descriptorの情報から、参照している仮想ファイルを管理する全てのVirtual File DescriptorのArea Management Space内での記録位置を把握し、全てのVirtual File Descriptorを読み出す。

【０１９１】ステップS130において、読み込んだVirtual File Descriptor内のPointer toParent Directoryによって削除しようとする仮想ファイルが含まれるディレクトリを管理するDirectory Descriptorを削除しようとする仮想ファイル分読み出す。

【０１９２】ステップS131において、読み込んだDirectory Descriptorから削除する仮想ファイルへのポインタであるDescriptor Pointer Entryを削除して、Directory Descriptorをディスク上で更新する。

【０１９３】ステップS132において、削除対象となった全ての仮想ファイルについて仮想プログラムファイルから参照を受けていたかどうかをチェックする。これは、上述したように仮想ファイルを削除する場合に、その仮想ファイルを参照している仮想プログラムファイルとの関係をチェックする必要があるからである。

【０１９４】すべてチェックが終わったら、ステップS127において、削除するオリジナルファイルが含まれるディレクトリを管理するDirectory Descriptorから削除するFile Descriptorへのポインタ情報であるDescriptor Pointer Entryを削除し、Directory Descriptorをディスク上で更新する。

【０１９５】ステップS128において、削除したオリジナルシーンの論理ファイルシステムの管理情報であるFile Descriptorと、Extent Space内の削除する映像データと、オリジナルシーンを参照していた仮想ファイルの論理ファイルシステムの管理情報であるVirtual File Descriptorおよび仮想ファイルを参照していた仮想プログラムファイルを管理するVirtual Program Descriptorの記録位置に対応するSpace Management DescriptorのSpace bitmapを開放し処理を終了する。

【０１９６】ステップS132において削除対象となった全ての仮想ファイルについて仮想プログラムファイルから参照を受けていたかどうかをチェックをし、もしすべて

チェックが終わっていなければ、ステップS133において、注目している仮想ファイルが仮想プログラムファイルから参照されているかどうかをVirtual File Descriptor内のNumber of References by Virtual Program Fileによって判断する。ステップS133において参照されていないと判断された場合は、ステップ132に戻り処理を繰り返す。

【０１９７】ステップS133において、仮想ファイルが仮想プログラムファイルから参照されていたら、ステップS134において、参照している全ての仮想プログラムファイルの管理情報であるVirtual Program File Descriptorを、Area Management Spaceから読み出す。この際、Virtual File Descriptor内のLocation of Virtual Program File Descriptorを利用する。ステップS135において、読み出した全ての仮想プログラムファイルの管理情報であるVirtual Program File Descriptor内のNumber of Referenced Virtual Fileの値をデクリメントする。

【０１９８】ステップS136において、同じくVirtual Program File Descriptor内のLocation of Virtual File Descriptorから削除する仮想ファイルの記録されていた位置情報を削除する。ステップS137において、ステップS135、S136変更があったVirtual Program File Descriptorをディスク上で更新し、ステップS132に戻り処理を繰り返す。

【０１９９】ステップS123において、仮想ファイルから参照されていない部分のみ削除する処理が選択された場合は、ステップS124において、削除しようとするオリジナルシーンを参照している仮想ファイルを管理するVirtual File Descriptorを、すべてArea Management Spaceより読み出す。この際、ステップS121において読み出したFile Descriptor内のNumber of References by Virtual File Descriptorと、Location of Virtual File Descriptorを利用する。

【０２００】ステップS125において、読み出した全ての仮想ファイルの管理情報であるVirtual File Descriptor内のNumber of ExtentsとLocation of Extent、実ファイルの管理情報であるFile Descriptor内のNumber of ExtentsとLocation of Extentを比較することによって、オリジナルシーンのどの部分が仮想ファイルから参照を受けていない部分かどうかを把握する。

【０２０１】ステップS126において、オリジナルシーンを管理するFile DescriptorのNumber of ExtentsおよびLocation of Extentで、仮想ファイルが参照している全ての参照箇所を含めるように、オリジナルシーンが管理するExtent Spaceの位置情報を更新する。ステップS128において、オリジナルシーンの内、仮想ファイルから参照を受けていなかった部分に対応するSpace Management DescriptorのSpace bitmapを更新して処理を終了する。

【０２０２】以上、オリジナルシーンが管理するファイルとそれを参照している仮想ファイルの参照情報について、論理ファイルシステムの管理情報として扱う場合に関して説明を行なって来た。実施形態１および２の例では論理ファイルシステムレベルで参照情報を管理しているので、デバイスドライバが前述の処理をすべて行う事が可能となり、ユーザシステムからはコマンドだけでディスクにアクセスすることができる。

【０２０３】次に、第3の実施形態について説明する。上記第1及び第2の実施形態で説明してきた、実ファイルと仮想ファイルの参照情報、つまり、仮想ファイルがどの実ファイルを参照しているかを示す参照情報を、論理ファイルシステムの上位レベルのユーザシステムがExtent Spaceに記録するファイルとして扱うことも考えられる。この場合は、論理ファイルシステムの管理情報としてはファイルおよび仮想ファイルの仕組みだけを用意することになる。この実施形態を実施形態3として説明を進める。

【０２０４】ファイルとそれを参照する仮想ファイルの仕組みは論理ファイルシステムに残されているので、ディスク間のオリジナルシーンやユーザシーンのコピーや、IEEE1394などのネットワークを利用して転送を行なったりする場合に、単純に論理ファイルシステムのファイルやディレクトリを指定してコピーや転送を行なうことによって、目的の事が達成される。オリジナルシーンやオリジナルプログラムの削除や変更を行なう場合は、編集アプリケーションプログラムを用いて、参照情報が記録されたファイルを読み出すことによって、参照関係を容易に把握することが可能となり、実施形態１および２で説明した場合と同じような処理が可能となる。さらに、File Descriptorの構成が実ファイル、仮想ファイルで共通の構成を取ることができる。

【０２０５】図37に実施形態3の概要を示す。基本的には、これまで説明してきた実施形態１と同様の構造であるが、新たに管理情報を保存するためのディレクトリMANGが定義されている点が異なる。このディレクトリの下にREF.MANというファイルが記録される。このファイルには、図37に示すように、ユーザシーンを管理する仮想ファイルと、オリジナルシーンを管理するファイルの間の参照関係が示されている。オリジナルシーンを削除しようと考えた場合、上記参照情報が記述されたREF.MANを編集アプリケーションプログラムが読み込んで実際の削除や変更作業を行なうとともに、参照情報が変更される。また、ユーザシーンを削除する場合にも、この参照情報が変更されることになる。

【０２０６】実施形態3においては、オリジナルシーンを管理するFile Descriptorとユーザシーンを管理するVirtual File Descriptorという管理情報の構成はとらず、1つのFile Descriptorによって両方を管理し、Attributeにより両者を区別するものとする。

【０２０７】図38に示すFile Descriptor (FD) はファ

イルを管理するための管理情報である。FDはFDを識別するためのID（Header ID）、１論理ブロックにFDが格納できずに複数の論理ブロックにまたがって記録される場合に次にアクセスすべき論理フロック番号（Next Extension）、同様に複数論理ブロックにまたがった場合に１つ前の情報が記録された論理ブロック番号（Previous Extension）、ファイルの属性情報（Attribute）、ファイル名（File Name）、 File Descriptorの作成および修正時刻（Creation Time & Date、 Modified Time & Date）、ファイルの大きさ（File Size）、管理するファイルが含まれるディレクトリのDirectoryDescriptorの記録された論理ブロック番号（Pointer to Parent Directory）、ファイル構造でディスク上の連続領域を管理する場合の位置情報（Location ofContiguous Extent）、ファイルあるいは仮想ファイルに対応するディスク上の配置の分断数（Number of Extents）、そしてそれぞれの分断の位置情報（Location of Extents）で構成される。

【０２０８】図39にファイルの属性情報であるAttributeについて示す。Attributeは16bitの情報であり、Bit0 Read Onlyは管理するファイルが読み込み専用であるかを示し、Bit1 Deletedは管理するファイルが一時的に削除されたかどうかを示し、Bit2 Contiguousは管理するファイルがディスク上の連続領域を確保しているかどうかを示し、Bit3 Allocation Modeはファイルで管理されるデータの配置方法がAreaの前方からアクセスであるか後方からのアクセスであるかを示し、Bit4 Virtual Modeは管理されるファイルが仮想ファイルであるかを示し、Bit5、6 CGMS（Copy Generation Management System）は、コピーを許可するか、1世代のみコピーするか、コピーを禁止するかどうかの情報を示す。なおBit7-15までは将来の拡張用にReservedされたビットである。

【０２０９】このように論理ファイルシステムにおいて、オリジナルシーンを管理するファイルとユーザシーンを管理する仮想ファイルの管理情報の構造は共通であり、Attribute情報内のBit4のVirtual Mode情報によって0の場合は実ファイル、1の場合は仮想ファイルといった方法で区別する。実際のユーザシーンの作成方法や削除などの方法は既に述べた実施形態１の場合と参照情報に関する情報がファイルとして保存される事以外は基本的に同じであるため動作説明は省略する。

【０２１０】以上のように、オリジナルシーンをファイルとして管理し、そのオリジナルシーンのデータを参照する形で定義されるユーザシーンを管理する仮想ファイルに関して、オリジナルシーンやユーザプログラムを消去したり変更したりする際に、ファイルと仮想ファイルの参照情報を論理ファイルシステムの管理情報として持つ場合と、編集アプリケーションが使用することを想定し、参照情報をファイルとして管理する方法について述べた。

【０２１１】いずれの実施形態においても、ユーザプログラムやユーザシーンといった管理単位が仮想的なファイルとして管理されているので、ユーザシーンやユーザプログラムのコピーや、図40に示すようなネットワーク経由の転送が容易に行なえることが可能となる。

【０２１２】また前者の論理ファイルシステムに参照情報を入れることによって、例えば、ネットワークを利用した遠隔操作を考えた場合、論理ファイルシステムレベルのコマンドである、COPY、MOVE、RENAME、DELETEなどの抽象化されたコマンドをデバイスドライバに対して行うことによって、ユーザシーンやユーザプログラムの簡易的な編集が可能となる。一方、参照情報をファイルとして保存する場合には、論理ファイルシステムのみを使って簡易的な編集はできないが、編集を行なうアプリケーションプログラムを使うことによって、参照情報の管理情報を読み込んで同様の事を達成することが可能となる。

【０２１３】また、実ファイルと仮想ファイルの参照関係を示す管理情報を用意することによって、オリジナルシーンを削除した場合に、そのオリジナルシーンを参照している仮想ファイルが参照しているデータが無くなってしまうことを防ぐだけではなく、オリジナルシーンを参照しているユーザシーンを管理する仮想ファイルも含めて削除してり、あるいは、削除しようとするオリジナルシーンのデータのうちユーザシーンから参照されていない部分のみ削除して不要なデータをディスクから削除したりすることが可能となる。

【０２１４】また、ファイルと仮想ファイルの参照情報が用意されているので、オリジナルシーンを参照しているユーザシーンを容易に特定することができ、映像の管理が楽に行えることになる。

【０２１５】

【発明の効果】本発明の第１の発明によれば、実ファイルの管理情報中に、該実ファイルを参照している仮想ファイルを識別するための情報を有することによって、例えば実ファイルを削除しようとした場合に、該実ファイルを参照している仮想ファイルがあるか否かを容易に判断することが可能となり、誤って、参照している仮想ファイルがあるにもかかわらず、実ファイルを削除してしまうということを防止するための構成を容易に構築できる。

【０２１６】また、該実ファイルと仮想ファイルの参照関係を共有情報として一つのファイルとして記録するように構成することによって、実ファイルと仮想ファイルの管理情報の構成を同一にすることが可能となる。

【０２１７】本発明の第２の発明によれば、仮想ファイルの管理情報中に、該仮想ファイルが参照している実ファイルを識別するための情報を有することによって、仮想ファイルを削除した場合など、該仮想ファイルが参照している実ファイルの管理情報を更新する必要が生じた

場合に、容易に該仮想ファイルが参照している実ファイルの管理情報を参照することができる。

【０２１８】本発明の第３の発明によれば、記録媒体上に記録された複数のファイル（実ファイル或いは仮想ファイル）を１つのプログラムとして管理し、該プログラムを管理するプログラム管理情報に、管理するファイルを識別する情報を備えることによって、例えば、ファイルを削除しようとした場合に該ファイルを含むプログラムがあるか否かを容易に判断することが可能となり、誤って、該ファイルを含むプログラムがあるにもかかわらず、該ファイルを削除してしまうということを防止するための構成を容易に構築できる。

【０２１９】本発明の第４の発明によれば、プログラムの管理情報中に、該プログラムに含まれるファイルを識別するための情報を有することによって、プログラムを削除した場合など、該プログラムが含むファイルの管理情報を更新する必要が生じた場合に、容易に該プログラムが含んでいるファイルの管理情報を参照することができる。

【図面の簡単な説明】

【図１】本発明のファイル管理方法の実施形態におけるシステム構成を示す説明図である。

【図２】本発明のファイル管理方法の実施形態におけるオリジナルシーンをディスクに記録した様子を示す図である。

【図３】本発明のファイル管理方法の実施形態で扱うMPEGストリームの構成を示す説明図である。

【図４】本発明のファイル管理方法の実施形態で扱うMPEGストリームとブロックの関係を示す説明図である。

【図５】本発明のファイル管理方法の実施形態において１つの仮想ファイルがオリジナルシーンのデータを参照している様子を示す説明図である。

【図６】本発明のファイル管理方法の実施形態において２つの仮想ファイルがオリジナルシーンのデータを参照している様子を示す説明図である。

【図７】本発明のファイル管理方法の実施形態におけるadr#long形式の説明図である。

【図８】本発明のファイル管理方法の実施形態におけるHeader ID形式の説明図である。

【図９】本発明のファイル管理方法の実施形態におけるPrimary Volume Descriptorの説明図である。

【図１０】本発明のファイル管理方法の実施形態におけるArea Descriptorの説明図である。

【図１１】本発明のファイル管理方法の実施形態におけるディスクのレイアウトを示す説明図である。

【図１２】本発明のファイル管理方法の実施形態におけるPrimary Area Descriptorの説明図である。

【図１３】本発明のファイル管理方法の実施形態におけるSpace Management Descriptorの説明図である。

【図１４】本発明のファイル管理方法の実施形態におけるDirectory Descriptorの説明図である。

【図１５】本発明のファイル管理方法の実施形態におけるDescriptor Pointer Entryの説明図である。

【図１６】本発明のファイル管理方法の実施形態におけるディレクトリ用のAttributeの説明図である。

【図１７】本発明のファイル管理方法の実施形態におけるExtended Descriptorの説明図である。

【図１８】本発明のファイル管理方法の実施形態におけるExtended Descriptorの使用方法を示す説明図である。

【図１９】本発明のファイル管理方法の実施形態におけるAV Areaのレイアウトを示す説明図である。

【図２０】本発明のファイル管理方法の実施形態におけるFile Descriptorの説明図である。

【図２１】本発明のファイル管理方法の実施形態におけるファイルのAttributeの説明図である。

【図２２】本発明のファイル管理方法の実施形態においてファイルを作成する際の処理の流れを示すフローチャートである。

【図２３】本発明のファイル管理方法の第１の実施形態におけるファイルと仮想ファイルの関係を示す説明図である。

【図２４】本発明のファイル管理方法の第１の実施形態におけるVirtual File Descriptorの説明図である。

【図２５】本発明のファイル管理方法の第１の実施形態における仮想ファイルのAttributeの説明図である。

【図２６】本発明のファイル管理方法の第１乃至２の実施形態において仮想ファイルを作成する処理の流れを示すフローチャートである。

【図２７】本発明のファイル管理方法の第１の実施形態において仮想ファイルを削除する処理の流れを示すフローチャートである。

【図２８】本発明のファイル管理方法の第１の実施形態においてファイルを削除する処理の流れを示すフローチャートである。

【図２９】本発明のファイル管理方法の第２の実施形態におけるファイルと仮想ファイルと仮想プログラムファイルの関係を示す説明図である。

【図３０】本発明のファイル管理方法の第２の実施形態における仮想プログラムファイルの応用例の関係を示す説明図である。

【図３１】本発明のファイル管理方法の第２の実施形態におけるVirtual File Descriptorの説明図である。

【図３２】本発明のファイル管理方法の第２の実施形態におけるVirtual Program File Descriptorの説明図である。

【図３３】本発明のファイル管理方法の第２の実施形態において仮想プログラムファイルを作成する処理の流れを示すフローチャートである。

【図３４】本発明のファイル管理方法の第２の実施形態

において仮想ファイルを削除する処理の流れを示すフローチャートである。

【図３５】本発明のファイル管理方法の第２の実施形態において仮想プログラムファイルを削除する処理の流れを示すフローチャートである。

【図３６】本発明のファイル管理方法の第２の実施形態においてファイルを削除する処理の流れを示すフローチャートである。

【図３７】本発明のファイル管理方法の第３の実施形態においてファイルと仮想ファイルの参照情報をファイルとし管理する様子を示すフローチャートである。

【図３８】本発明のファイル管理方法の第３の実施形態におけるFile Descriptorの説明図である。

【図３９】本発明のファイル管理方法の第３の実施形態におけるAttributeの説明図である。

【図４０】本発明のファイル管理方法の実施形態における

るネットワーク機器の接続の様子を示す説明図である。

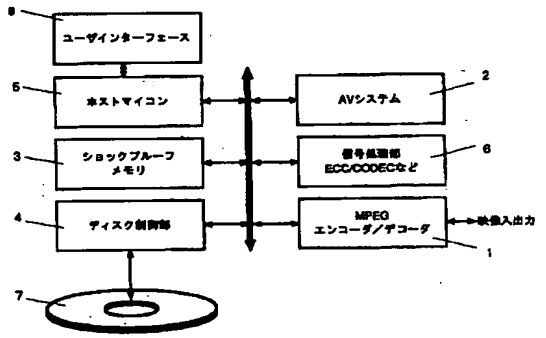
【図４１】従来技術におけるユーザシステムがデバイスドライバを使って論理ファイルシステムの管理情報を使ってディスクにアクセスする様子の説明図である。

【図４２】従来技術における仮想ファイルの説明図である。

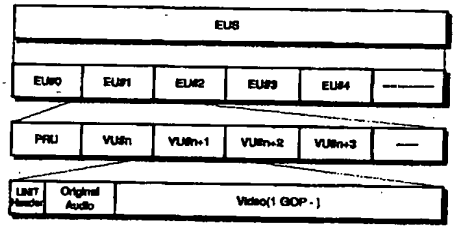
【符号の説明】

1　ＭＰＥＧエンコーダ／デコーダ
2　ＡＶシステム
3　ショックプルーフメモリ
4　ディスク制御部
5　ホストマイコン
6　信号処理部
7　記録媒体
8　ユーザインタフェース

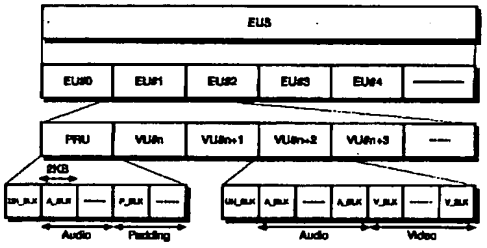
【図１】



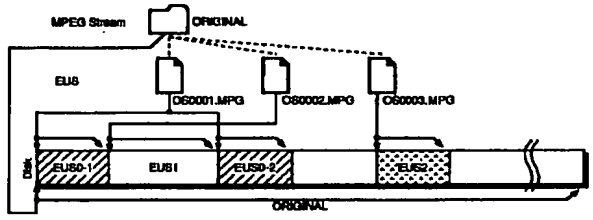
【図３】



【図４】



【図２】



【図７】

str_long

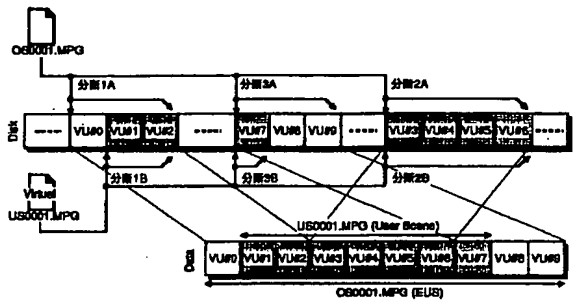
BP	Length	Field Name	Contents
0	4	Location	UInt32
4	4	Length	UInt32

【図８】

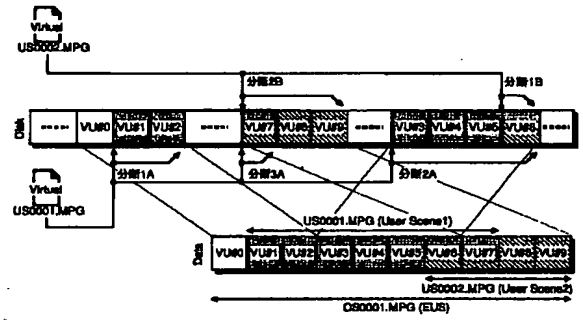
Header ID

BP	Length	Field Name	Contents
0	2	Standard ID	string="SA"
2	2	Standard Version	UInt16
4	2	Descriptor ID	UInt16
6	2	Manufacturer ID	UInt16

【図５】



【図６】



【図９】

Primary Volume Descriptor

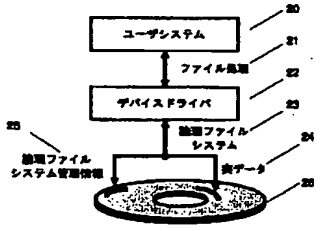
BP	Length	Field Name	Contents
0	8	Header ID	Header ID
8	1	Disk Type	UInt8
9	3	Reserved	all 00h
12	4	Disk ID	UInt32
16	4	Volume Size	UInt32
20	4	Free Logical Sectors in Allocatable Space	UInt 32
24	24	Volume Name	String
48	4	Creation Time & Date	Timestamp
52	4	Modified Time & Date	Timestamp
56	8	Location of Volume Management Space	adr_long
64	8	Location of Allocatable Space	adr_long
72	2	Number of Areas	UInt16(=NOA)
74	2	Reserved	0000h
76	NOA*4	Location of Area Descriptor	UInt32

【図１０】

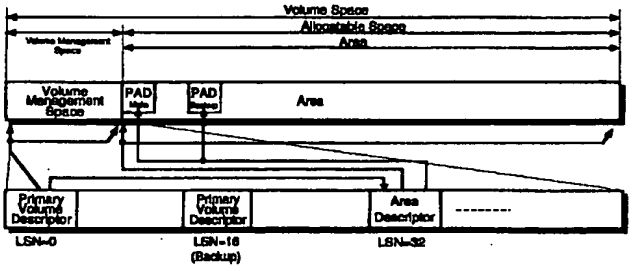
Area Descriptor

BP	Length	Field Name	Contents
0	8	Header ID	Header ID
8	1	Area Type	UInt8
9	3	Reserved	all 00h
12	4	Logical Block Size	UInt32
16	4	Area Size	UInt32
20	24	Area Name	string
44	4	Creation Time & Date	Timestamp
48	4	Modified Time & Date	Timestamp
52	4	Location of Main Primary Area Descriptor	UInt32
56	4	Location of Secondary Primary Area Descriptor	UInt32
60	2	Number of Area Extents	UInt16(=NOAE)
62	2	Reserved	0000h
64	NOAE*8	Location of Area Extent	adr_long

【図４１】



【図１１】



【図１３】

Space Management Descriptor

BP	Length	Field Name	Contents
0	8	Header ID	Header ID
8	4	Next Extension	UInt32
12	4	Previous Extension	UInt32
16	AreaSize/8	Space Bitmap	-

【図１５】

Descriptor Pointer Entry

BP	Length	Field Name	Contents
0	24	Entry Name	string
24	1	Entry Type	UInt8
25	3	Reserved	all 00h
28	4	Location of Descriptor	UInt32

【図１２】

Primary Area Descriptor

BP	Length	Field Name	Contents
0	8	Header ID	Header ID
8	8	Reserved	all 00h
16	4	Area Size	Uint32
20	4	Free Blocks in Area Management Space	Uint32
24	4	Free Blocks in Extent Space	Uint32
28	1	Area Mode	Uint8
29	3	Reserved	all 00h
32	4	Logical Block Size	Uint32
36	24	Area Name	string
60	4	Creation Time & Date	Timestamp
64	4	Modified Time & Date	Timestamp
68	8	Location of Main Area Management Space	adr_long
76	8	Location of Secondary Area Management Space	adr_long
84	8	Location of Extent Space	adr_long
92	4	Location of Space Management Descriptor	Uint32
96	4	Location of Root Directory Descriptor	Uint32

【図１４】

Directory Descriptor

BP	Length	Field Name	Contents
0	8	Header ID	Header ID
8	4	Next Extension	Uint32
12	4	Previous Extension	Uint32
16	2	Attribute	Uint16
18	2	Reserved	all 00h
20	24	Directory Name	string
44	4	Creation Time & Date	Timestamp
48	4	Modified Time & Date	Timestamp
52	4	Pointer to Parent Directory	Uint32
56	8	Location of Contiguous Space	adr_long
64	2	Number of Descriptor Pointer Entries	Uint16(=NODPE)
66	2	Reserved	0000h
68	NODPE*32	Descriptor Pointer Entry	-

【図１６】

Attribute (for Directory)

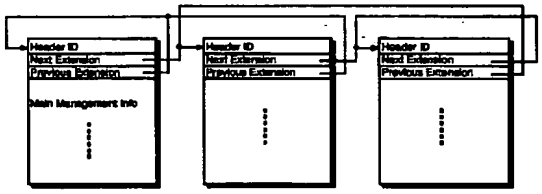
Bit	Description	Contents
0	Read Only	ZERO/ONE
1	Deleted	ZERO/ONE
2	Contiguous	ZERO/ONE
3	Allocation Mode	ZERO
4	CGMS	ZERO/ONE
5	CGMS	ZERO/ONE
6～15	Reserved	ZERO

【図１７】

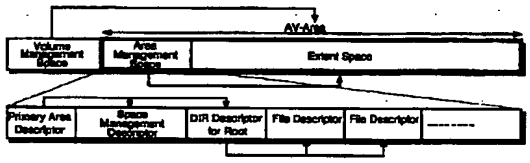
Extended Descriptor

BP	Length	Field Name	Contents
0	8	Header ID	Header ID
8	4	Next Extension	Uint32
12	4	Previous Extension	Uint32
16	-	Extended Data	-

【図１８】



【図１９】



【図２０】

File Descriptor

BP	Length	Field Name	Contents
0	8	Header ID	Header ID
8	4	Next Extension	Uint32
12	4	Previous Extension	Uint32
16	2	Attribute	Uint16
18	2	Reserved	0000h
20	24	File Name	string
44	4	Creation Time & Date	Timestamp
48	4	Modified Time & Date	Timestamp
52	8	File Size	Uint48
56	2	Reserved	0000h
60	4	Pointer to Parent Directory	Uint32
64	8	Location of Contiguous Extent	adr_long
72	2	Number of References by Virtual File	Uint16(=NORV)
74	2	Reserved	0000h
76	NORV*4	Location of Virutal File Descriptor	Uint32
-	2	Number of Extents	Uint16(=NOE)
-	2	Reserved	0000h
-	NOE*8	Location of Extent	adr_long

【図２１】

Attribute (for File)

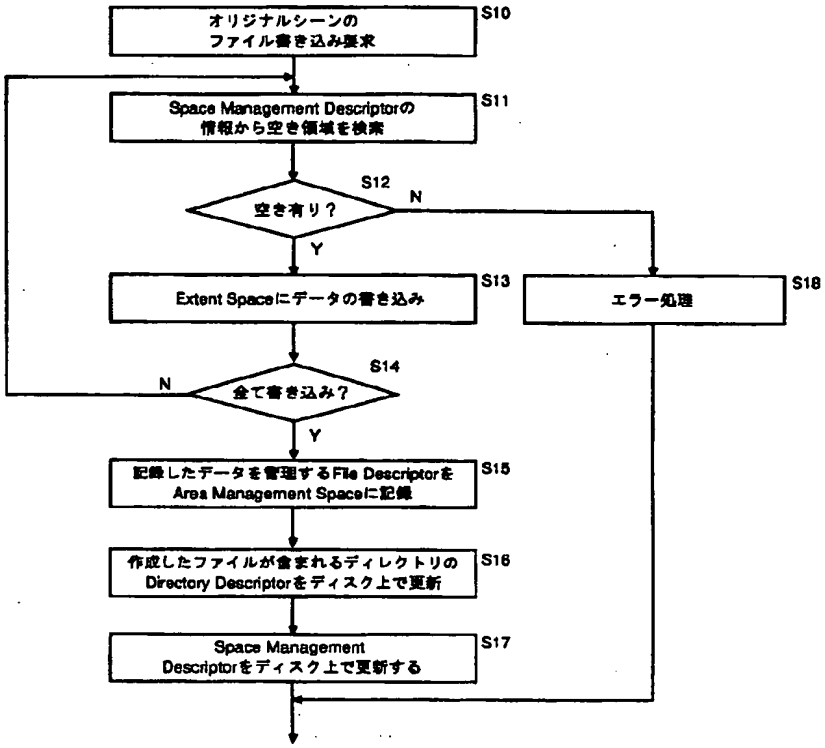
Bit	Description	Contents
0	Read Only	ZERO/ONE
1	Deleted	ZERO/ONE
2	Contiguous	ZERO/ONE
3	Allocation Mode	ZERO/ONE
4	CGMS	ZERO/ONE
5	CGMS	ZERO/ONE
6～15	Reserved	ZERO

【図２５】

Attribute (for Virtual File / Virtual Program File)

Bit	Description	Contents
0	Read Only	ZERO/ONE
1	Deleted	ZERO/ONE
2	Contiguous	ZERO
3	Allocation Mode	ZERO
4	CGMS	ZERO/ONE
5	CGMS	ZERO/ONE
6～15	Reserved	ZERO

【図２２】

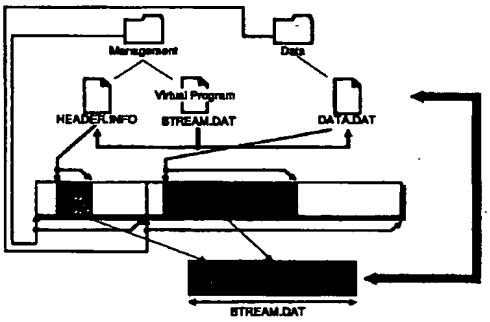


【図２４】

Virtual File Descriptor

BP	Length	Field Name	Contents
0	8	Header ID	Header ID
8	4	Next Extension	UInt32
12	4	Previous Extension	UInt32
16	2	Attribute	UInt16
18	2	Reserved	0000h
20	24	File Name	string
44	4	Creation Time & Date	Timestamp
48	4	Modified Time & Date	Timestamp
52	6	File Size	UInt48
58	2	Reserved	0000h
60	4	Pointer to Parent Directory	UInt32
64	4	Location of Referenced File Descriptor	UInt32
68	2	Number of Extents	UInt16(=NOE)
70	2	Reserved	0000h
72	NOE*8	Location of Extent	adr_long

【図３０】

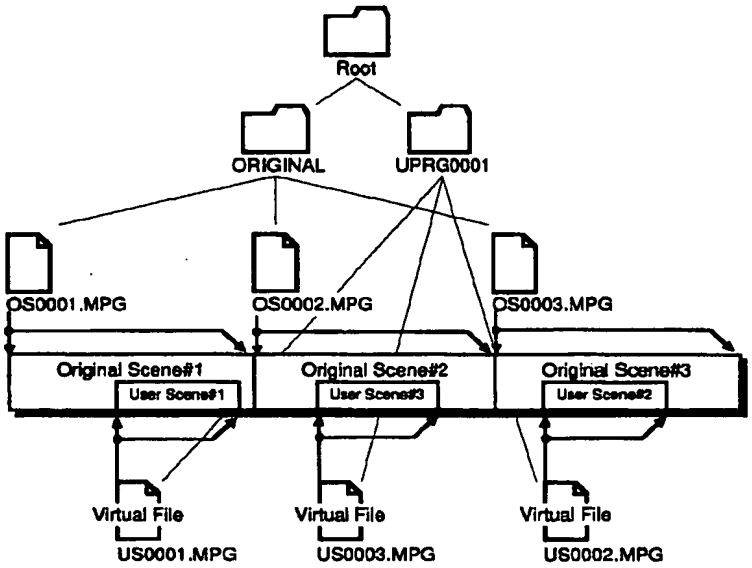


【図３９】

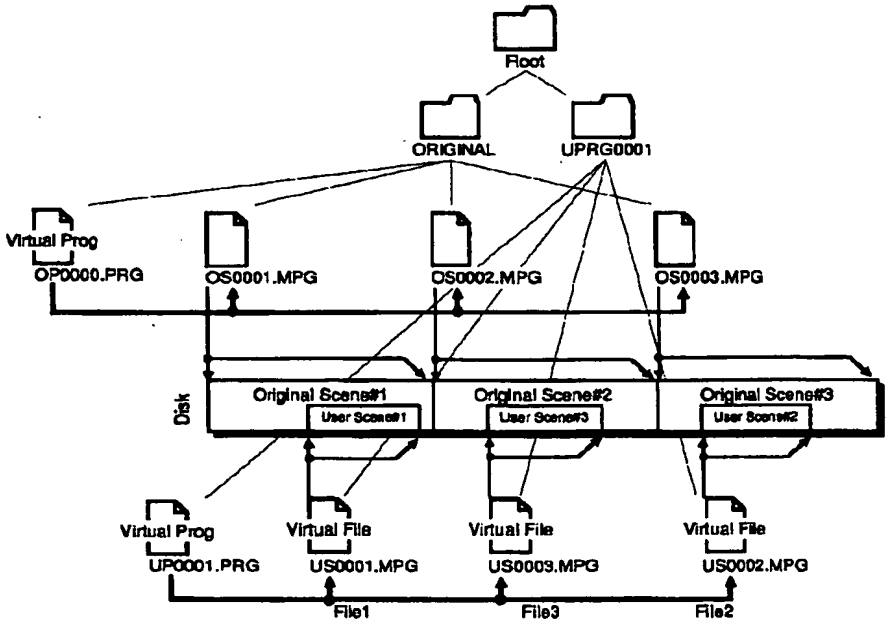
Attribute (for File)

Bit	Description	Contents
0	Read Only	ZERO/ONE
1	Deleted	ZERO/ONE
2	Contiguous	ZERO/ONE
3	Allocation Mode	ZERO/ONE
4	Virtual Mode	ZERO/ONE
5	CGMS	ZERO/ONE
6	CGMS	ZERO/ONE
7—15	Reserved	ZERO

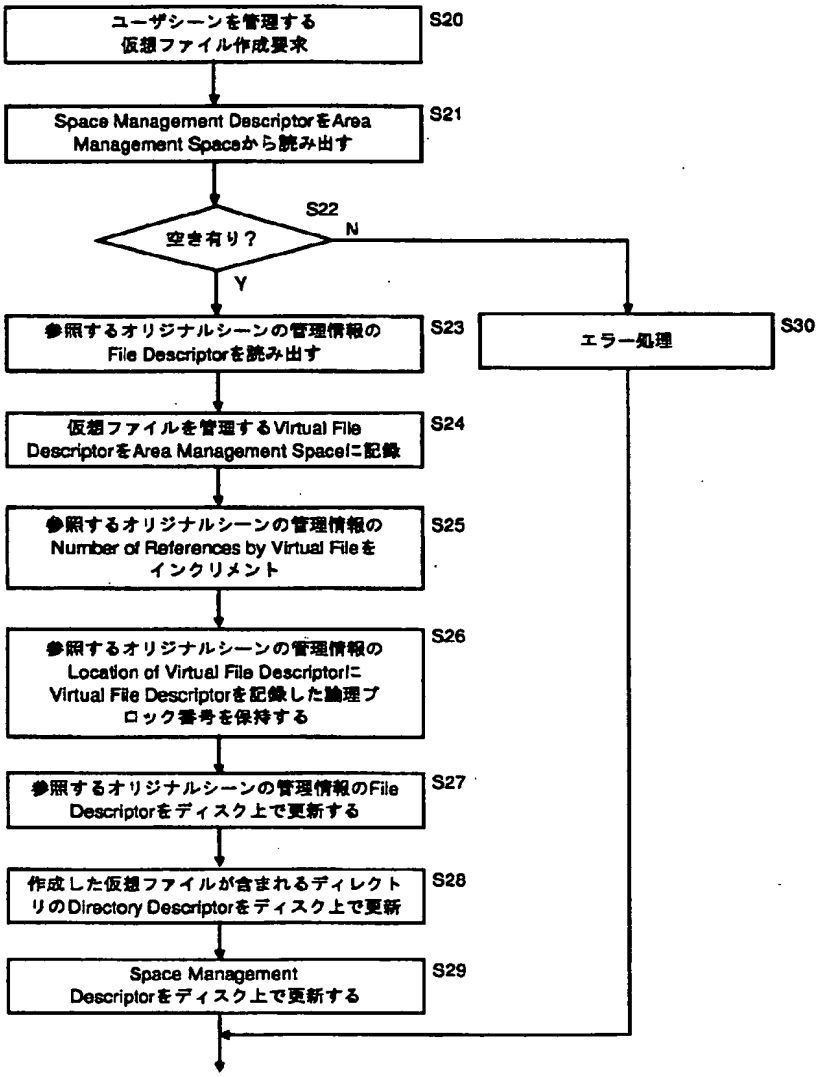
【図２３】



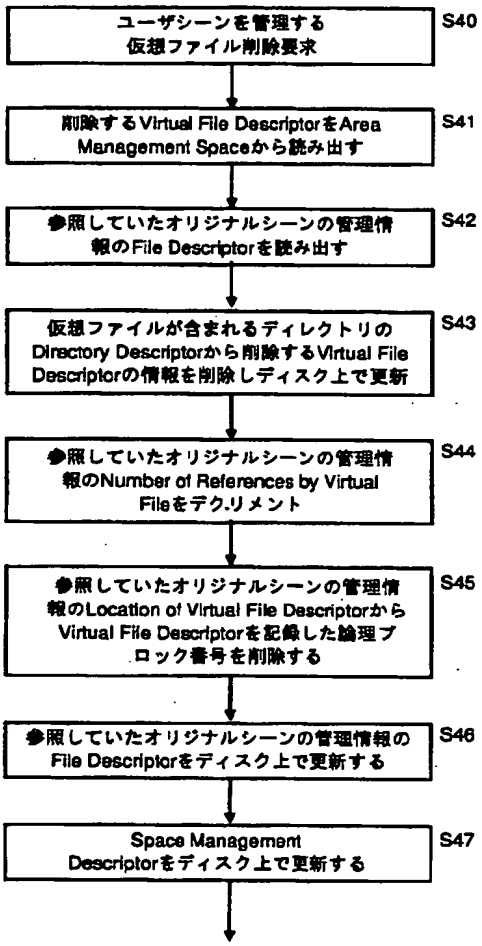
【図２９】



【図２６】



【図２７】



【図３１】

Virtual Program File Descriptor

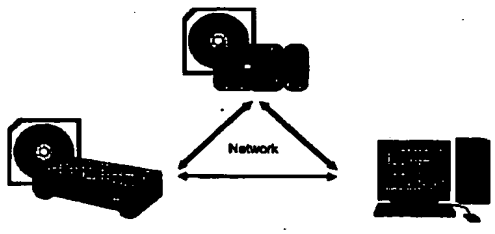
BP	Length	Field Name	Contents
0	8	Header ID	Header ID
8	4	Next Extension	Uint32
12	4	Previous Extension	Uint32
16	2	Attribute	Uint16
18	2	Reserved	0000h
20	24	File Name	string
44	4	Creation Time & Date	Timestamp
48	4	Modified Time & Date	Timestamp
52	6	File Size	Uint48
58	2	Reserved	0000h
60	4	Pointer to Parent Directory	Uint32
64	2	Number of References by Virutal File	Uint16(=NORVF)
66	2	Reserved	0000h
68	NORVF*4	Location of Virtual File Descriptor	Uint32

【図３８】

File Descriptor

BP	Length	Field Name	Contents
0	8	Header ID	Header ID
8	4	Next Extension	Uint32
12	4	Previous Extension	Uint32
16	2	Attribute	Uint16
18	2	Reserved	0000h
20	24	File Name	string
44	4	Creation Time & Date	Timestamp
48	4	Modified Time & Date	Timestamp
52	6	File Size	Uint48
58	2	Reserved	0000h
60	4	Pointer to Parent Directory	Uint32
64	8	Location of Contiguous Extent	adr_long
72	2	Number of Extents	Uint16(=NOE)
74	2	Reserved	0000h
76	NOE*8	Location of Extent	adr_long

【図４０】

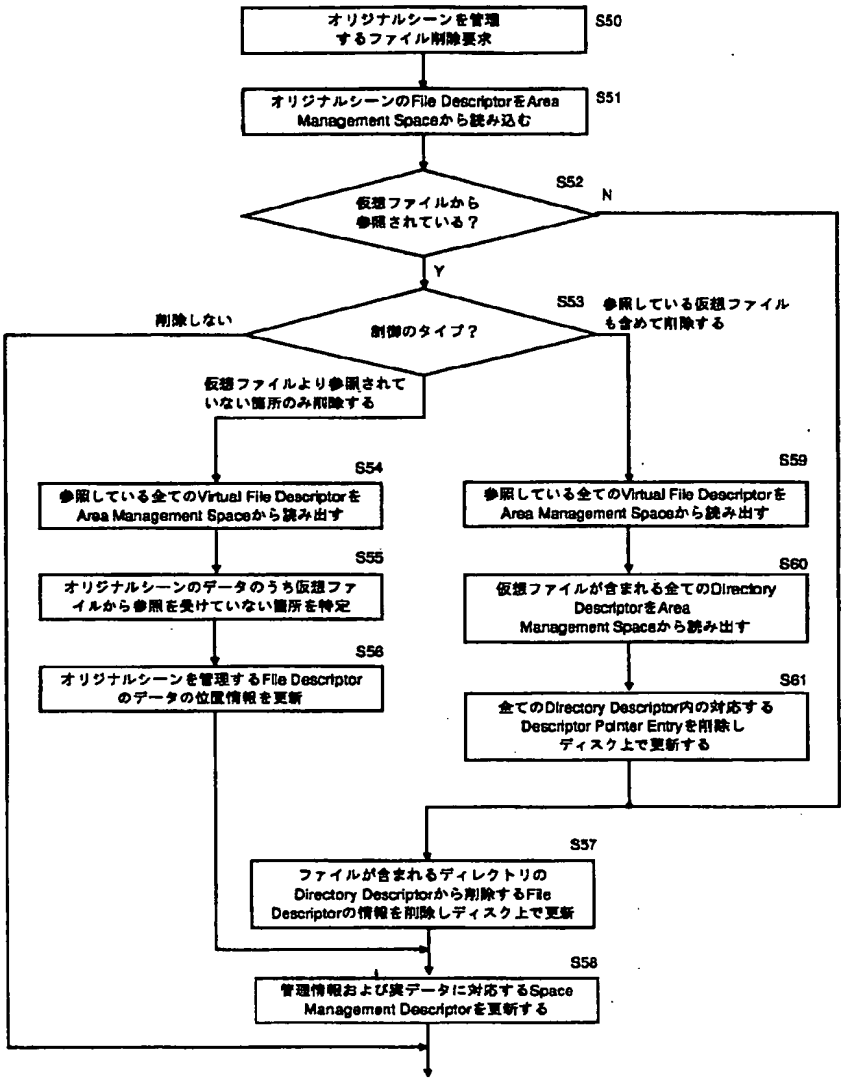


【図３２】

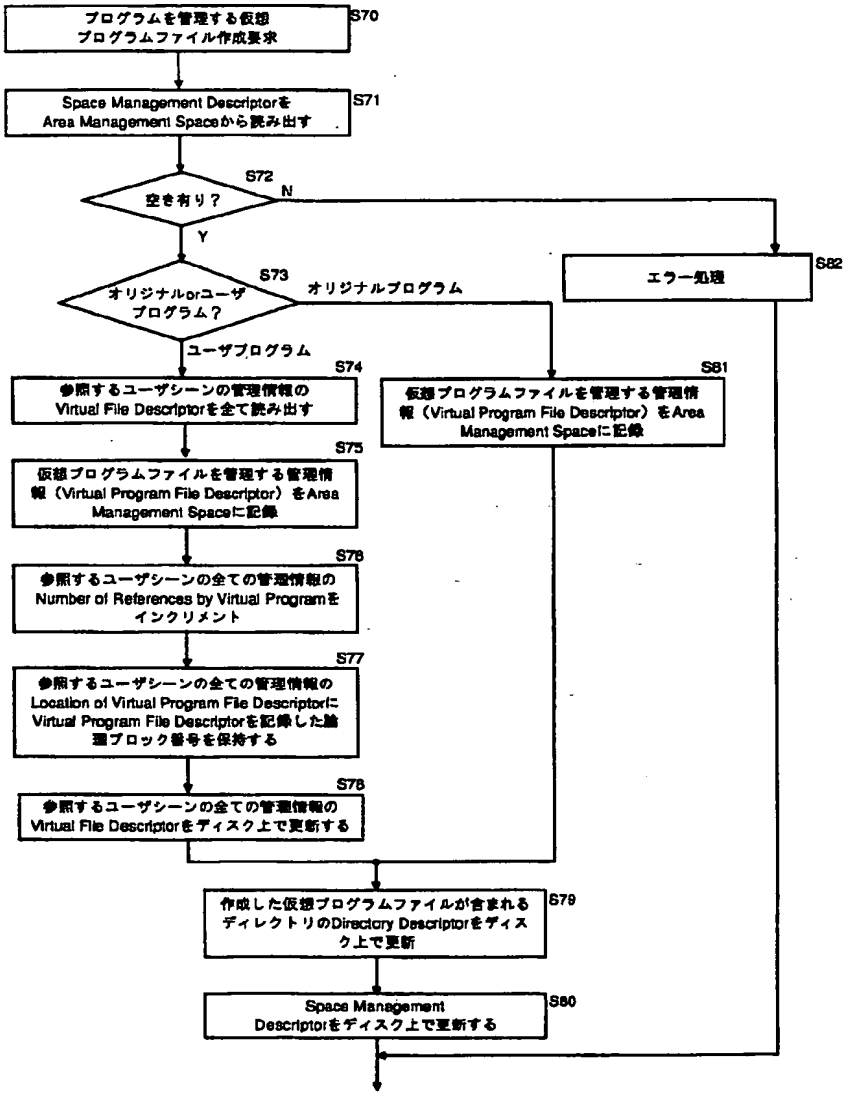
Virtual File Descriptor

BP	Length	Field Name	Contents
0	8	Header ID	Header ID
8	4	Next Extension	Uint32
12	4	Previous Extension	Uint32
16	2	Attribute	Uint16
18	2	Reserved	0000h
20	24	File Name	string
44	4	Creation Time & Date	Timestamp
48	4	Modified Time & Date	Timestamp
52	6	File Size	Uint48
58	2	Reserved	0000h
60	4	Pointer to Parent Directory	Uint32
64	4	Location of Referenced File Descriptor	Uint32
68	2	Number of References by Virtual Program	Uint16(=NORV)
70	2	Reserved	0000h
72	NORV*4	Location of Virtual Program File Descriptor	Uint32
-	2	Number of Extents	Uint16(=NOE)
-	2	Reserved	0000h
-	NOE*8	Location of Extent	adr_long

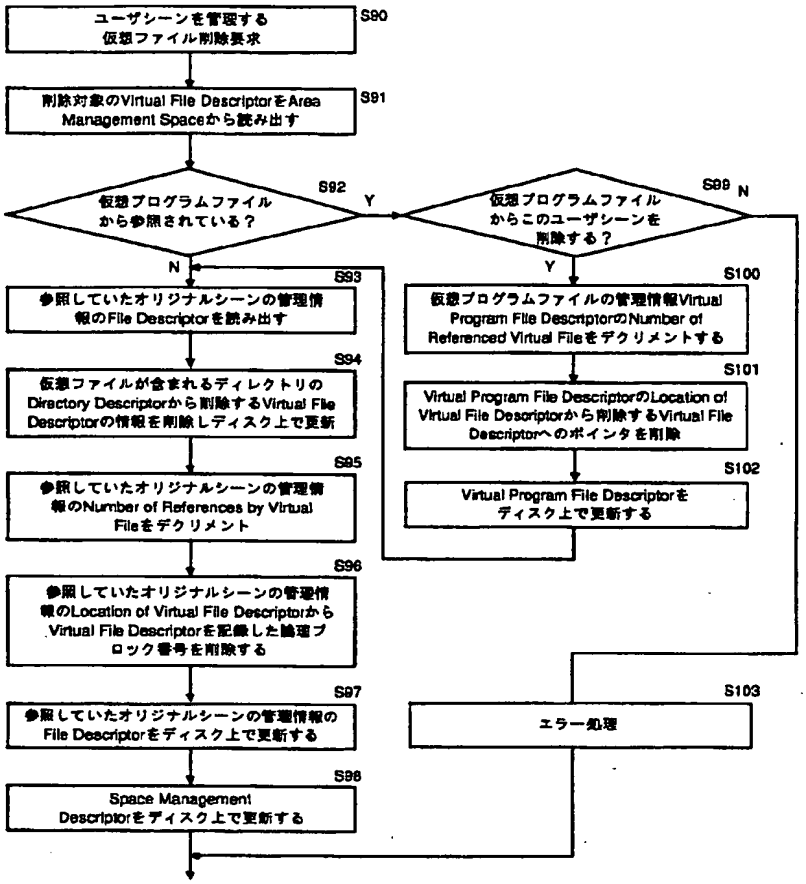
【図２８】



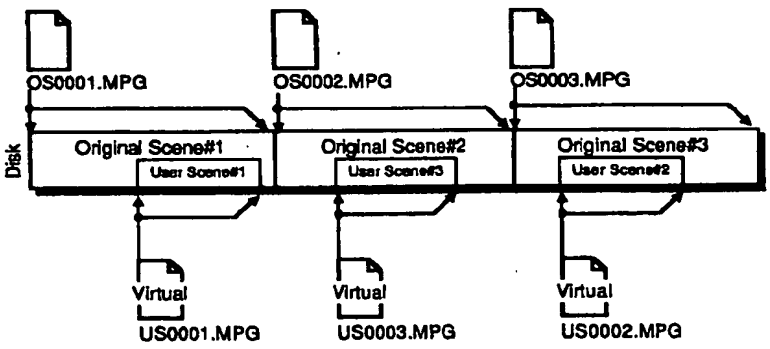
【図３３】



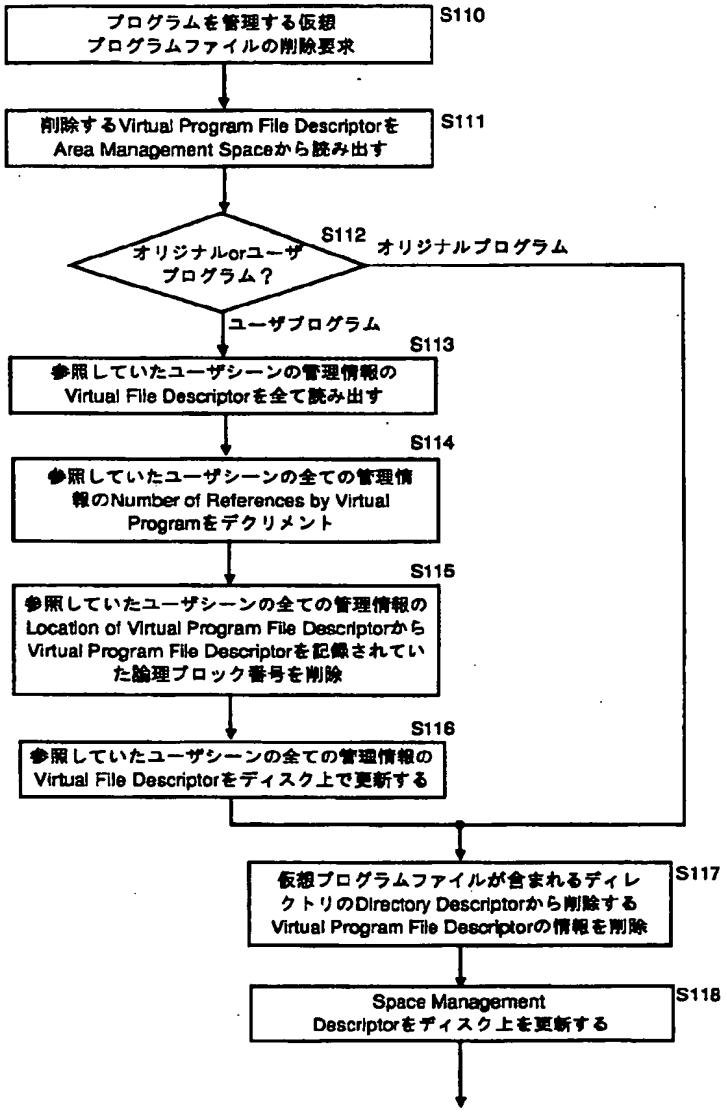
【図３４】



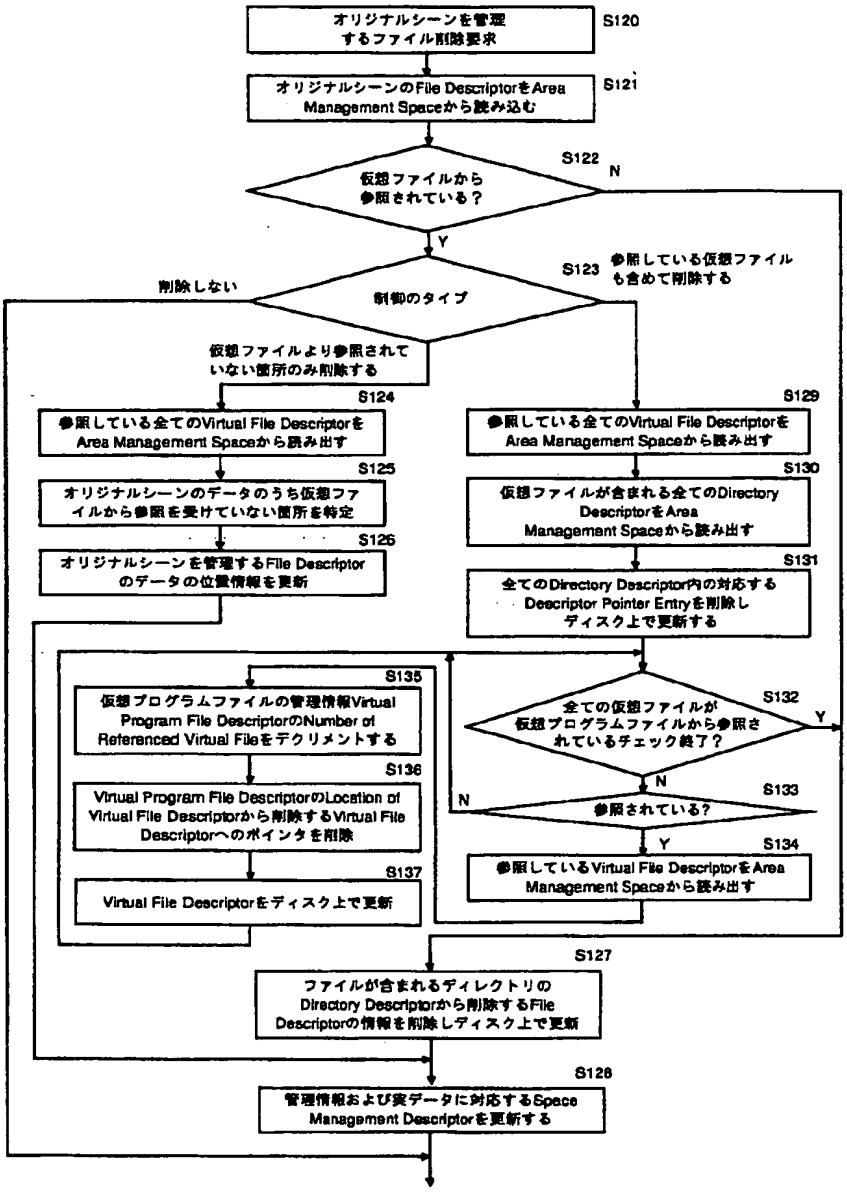
【図４２】



【図３５】



【図３６】



【図３７】